

# COMMUNICATIONS OF THE ACM

CACM.ACM.ORG

06/2009 VOL.52 NO.06

## One Laptop Per Child: Vision vs. Reality

Hard-Disk Drives:  
The Good, The Bad,  
and the Ugly

How CS Serves  
The Developing World

Network Front-End  
Processors

The Claremont Report  
On Database Research

Autonomous  
Helicopters



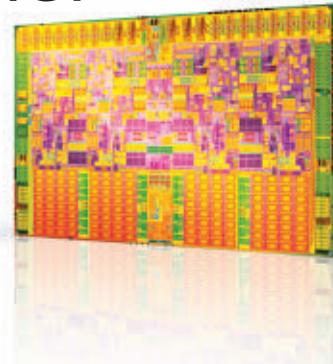


# Think Parallel....

It's not just what we make.  
It's what we make possible.

Advancing Technology Curriculum  
Driving Software Evolution  
Fostering Tomorrow's Innovators

Learn more at: [www.intel.com/thinkparallel](http://www.intel.com/thinkparallel)



# Noteworthy Computer Science Journals



## Autonomous Robots

G. Sukhatme, University of Southern California, Viterbi School of Engineering, Dept. Computer Science

**Autonomous Robots** reports on the theory and

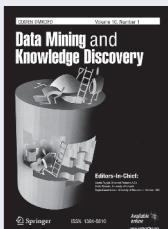
applications of robotic systems capable of some degree of self-sufficiency. It features papers that include performance data on actual robots in the real world. The focus is on the ability to move and be self-sufficient, not on whether the system is an imitation of biology. Of course, biological models for robotic systems are of major interest to the journal since living systems are prototypes for autonomous behavior.

► High Impact Factor in Robotics and AI

ISSN 0929-5593 (print version)

ISSN 1573-7527 (electronic version)

Journal no. 10514



## Data Mining and Knowledge Discovery

G. I. Webb, Monash University, School of Computer Science &, Software Engineering

The premier technical publication in the field,

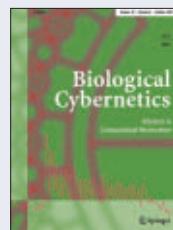
**Data Mining and Knowledge Discovery** is a resource collecting relevant common methods and techniques and a forum for unifying the diverse constituent research communities. The journal publishes original technical papers in both the research and practice of data mining and knowledge discovery, surveys and tutorials of important areas and techniques, and detailed descriptions of significant applications.

► High Impact Factor in Information Systems and AI

ISSN 1384-5810 (print version)

ISSN 1573-756X (electronic version)

Journal no. 10618



## Biological Cybernetics

W. Senn, Universität Bern, Physiologisches Institut; J. Rinzel, National Institutes of Health (NIH), Dept. Health Education & Welfare; J. L. van Hemmen, TU München, Abt. Physik

**Biological Cybernetics** is an interdisciplinary medium for experimental, theoretical and application-oriented aspects of information processing in organisms, including sensory, motor, cognitive, and ecological phenomena. Under the main aspects of performance and function of systems, emphasis is laid on communication between life sciences and technical/theoretical disciplines.

ISSN 0340-1200 (print version)

ISSN 1432-0770 (electronic version)

Journal no. 422



## Personal and Ubiquitous Computing

ACM

P. Thomas, Univ. Coll. London Interaction Centre

Personal and Ubiquitous

Computing publishes peer-reviewed international research on handheld, wearable and mobile information devices and the pervasive communications infrastructure that supports them to enable the seamless integration of technology and people in their everyday lives. The journal carries compellingly-written, timely and accessible contributions that illuminate the technological, social and design challenges of personal and ubiquitous computing technologies.

ISSN 1617-4909 (print version)

ISSN 1617-4917 (electronic version)

Journal no. 779

## Scientometrics

T. Braun, Lorand Eötvös University, Inst. Inorganic and Analytical Chemistry

**Scientometrics** is concerned with the quantitative features and characteristics of science. Emphasis is placed on investigations in which the development and mechanism of science are studied by statistical mathematical methods. The journal publishes original studies, short communications, preliminary reports, review papers, letters to the editor and book reviews on scientometrics.

► High Impact Factors in Computer Science., Interdisciplinary Applications and Information Science & library Science

ISSN 0138-9130 (print version)

ISSN 1588-2861 (electronic version)

Journal no. 11192



## Cybernetics and Systems Analysis

I. V. Sergienko, Acad. Science Ukraine, Glushkov Institute Cybernetics

**Cybernetics and System Analysis** publishes articles on: software and

hardware; algorithm theory and languages; programming and programming theory; optimization; operations research; digital and analog methods; hybrid systems; machine-machine and man-machine interfacing.

ISSN 1060-0396 (print version)

ISSN 1573-8337 (electronic version)

Journal no. 10559

# COMMUNICATIONS OF THE ACM

## Departments

- 5 **ACM-W Letter**  
**ACM-W Celebrates Women in Computing**  
By Elaine Weyuker

- 9 **Letters To The Editor**  
**Share the Threats**

- 10 **blog@CACM**  
**Speech-Activated User Interfaces and Climbing Mt. Exascale**  
Tessa Lau discusses why she doesn't use the touch screen on her in-car GPS unit anymore and Daniel Reed considers the future of exascale computing.

- 12 **CACM Online**  
**Making That Connection**  
By David Roman

- 27 **Calendar**

- 101 **Careers**

## Last Byte

- 103 **Puzzled**  
**Solutions and Sources**  
By Peter Winkler

- 104 **Future Tense**  
**Webmind Says Hello**  
By Robert J. Sawyer

## News



- 13 **Micromedicine to the Rescue**  
Medical researchers have long dreamed of "magic bullets" that go directly where they are needed. With micromedicine, this dream could become a life-saving reality.  
By Don Monroe

- 16 **Content Control**  
Entertainment businesses say digital rights management prevents the theft of their products, but access control technologies have been a uniform failure when it comes to preventing piracy. Fortunately, change is on the way.  
By Leah Hoffmann

- 18 **Autonomous Helicopters**  
Researchers are improving unmanned helicopters' capabilities to address regulatory requirements and commercial uses.  
By Gregory Goth

- 21 **Looking Backward and Forward**  
CRA's Computing Community Consortium hosted a day-long symposium to discuss the important computing advances of the last several decades and how to sustain that track record of innovation.  
By Bob Violino

## Viewpoints

- 22 **Privacy and Security**  
**Answering the Wrong Questions Is No Answer**  
Asking the wrong questions when building and deploying systems results in systems that cannot be sufficiently protected against the threats they face.  
By Eugene H. Spafford

- 25 **Inside Risks**  
**Reducing Risks of Implantable Medical Devices**  
A prescription to improve security and privacy of pervasive health care.  
By Kevin Fu

- 28 **The Profession of IT**  
**Beyond Computational Thinking**  
If we are not careful, our fascination with "computational thinking" may lead us back into the trap we are trying to escape.  
By Peter J. Denning

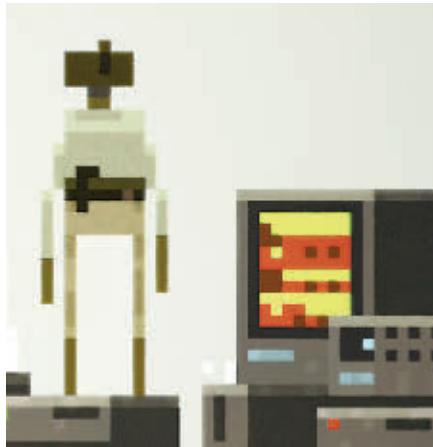
- 31 **Viewpoint**  
**Why "Open Source" Misses the Point of Free Software**  
Decoding the important differences in terminology, underlying philosophy, and value systems between two similar categories of software.

By Richard Stallman

- 34 **Kode Vicious**  
**Obvious Truths**  
How to determine when to put the brakes on late-running projects and untested software patches.  
By George V. Neville-Neil



Association for Computing Machinery  
Advancing Computing as a Science & Profession

**Practice****38 Hard-Disk Drives: The Good, the Bad, and the Ugly**

New drive technologies and increased capacities create new categories of failure modes that will influence system designs.

*By Jon Elerath*

**46 Network Front-end Processors, Yet Again**

The history of NFE processors sheds light on the trade-offs involved in designing network stack software.

*By Mike O'Dell*

**51 Whither Sockets?**

High bandwidth, low latency, and multihoming challenge the sockets API.

*By George V. Neville-Neil*

 Article development led by **queue.acm.org**



**About the Cover:**  
The One Laptop Per Child vision is being overwhelmed by the reality of business, politics, logistics, and competing interests worldwide.

The photo illustration on the cover is adapted from OLPC photos taken in the Gobi Desert.

**Contributed Articles****56 The Claremont Report on Database Research**

*By Rakesh Agrawal, Anastasia Ailamaki, Philip A. Bernstein, Eric A. Brewer, Michael J. Carey, Surajit Chaudhuri, AnHai Doan, Daniela Florescu, Michael J. Franklin, Hector Garcia-Molina, Johannes Gehrke, Le Gruenwald, Laura M. Haas, Alon Y. Halevy, Joseph M. Hellerstein, Yannis E. Ioannidis, Hank F. Korth, Donald Kossmann, Samuel Madden, Roger Magoulas, Beng Chin Ooi, Tim O'Reilly, Raghu Ramakrishnan, Sunita Sarawagi, Michael Stonebraker, Alexander S. Szalay, and Gerhard Weikum*

**66 One Laptop Per Child: Vision vs. Reality**

*By Kenneth L. Kraemer, Jason Dedrick, and Prakul Sharma*

**Review Articles****74 How Computer Science Serves the Developing World**

*By M. Bernardine Dias and Eric Brewer*

**Research Highlights****82 Technical Perspective****Reframing Security for the Web**

*By Andrew Myers*

**83 Securing Frame Communication in Browsers**

*By Adam Barth, Collin Jackson, and John C. Mitchell*

**92 Technical Perspective****Software and Hardware Support for Deterministic Replay of Parallel Programs**

*By Norman P. Jouppi*

**93 Two Hardware-based Approaches for Deterministic Multiprocessor Replay**

*By Derek R. Howe, Pablo Montesinos, Luis Ceze, Mark D. Hill, and Josep Torrellas*

**Virtual Extension**

As with all magazines, page limitations often prevent the publication of articles that might otherwise be included in the print edition. To ensure timely publication, ACM created *Communications*' Virtual Extension (VE).

VE articles undergo the same rigorous review process as those in the print edition and are accepted for publication on their merit. These articles are now available to ACM members in the Digital Library.

**Deriving Mutual Benefits from Offshore Outsourcing**

*Amar Gupta*

**Advancing Information Technology in Health Care**

*Steven M. Thompson and Matthew D. Dean*

**The Challenge of Epistemic Divergence in IS Development**

*Mark Lycett and Chris Partridge*

**Hyperlinking the Work for Self-Management of Flexible Workflows**

*Jonghun Park and Kwanho Kim*

**Re-Tuning the Music Industry—Can They Re-Attain Business Resonance?**

*Sudip Bhattacharjee, Ram D. Gopal, James R. Marsden, and Ramesh Sankaranarayanan*

**A Holistic Framework for Knowledge Discovery and Management**

*Dursun Delen and Suliman Al-Hawamdeh*

**Forensics of Computers and Handheld Devices: Identical or Fraternal Twins?**

*Nena Lim and Anne Khoo*

**Technical Opinion****Leveraging First-Mover Advantages in Internet-based Consumer Services**

*T.P. Liang, Andrew J. Czaplewski, Gary Klein, and James J. Jiang*



# COMMUNICATIONS OF THE ACM

A monthly publication of ACM Media

*Communications of the ACM* is the leading monthly print and online magazine for the computing and information technology fields. *Communications* is recognized as the most trusted and knowledgeable source of industry information for today's computing professional. *Communications* brings its readership in-depth coverage of emerging areas of computer science, new trends in information technology, and practical applications. Industry leaders use *Communications* as a platform to present and debate various technology implications, public policies, engineering challenges, and market trends. The prestige and unmatched reputation that *Communications of the ACM* enjoys today is built upon a 50-year commitment to high-quality editorial content and a steadfast dedication to advancing the arts, sciences, and applications of information technology.

ACM, the world's largest educational and scientific computing society, delivers resources that advance computing as a science and profession. ACM provides the computing field's premier Digital Library and serves its members and the computing profession with leading-edge publications, conferences, and career resources.

**Executive Director and CEO**  
John White  
**Deputy Executive Director and COO**  
Patricia Ryan  
**Director, Office of Information Systems**  
Wayne Graves  
**Director, Office of Financial Services**  
Russell Harris  
**Director, Office of Membership**  
Lillian Israel  
**Director, Office of SIG Services**  
Donna Cappo

**ACM COUNCIL**  
**President**  
Wendy Hall  
**Vice-President**  
Alain Chesaïnais  
**Secretary/Treasurer**  
Barbara Ryder  
**Past President**  
Stuart I. Feldman  
**Chair, SGB Board**  
Alexander Wolf  
**Co-Chairs, Publications Board**  
Ronald Boisvert, Holly Rushmeier  
**Members-at-Large**  
Carlo Ghezzi;  
Anthony Joseph;  
Mathai Joseph;  
Kelly Lyons;  
Bruce Maggs;  
Mary Lou Soffa;  
**SGB Council Representatives**  
Norman Jouppi;  
Robert A. Walker;  
Jack Davidson

**PUBLICATIONS BOARD**  
**Co-Chairs**  
Ronald F. Boisvert and Holly Rushmeier  
**Board Members**  
Gul Agha; Michel Beaudouin-Lafon;  
Jack Davidson; Nikil Dutt; Carol Hutchins;  
Ee-Peng Lim; M. Tamer Ozsu; Vincent  
Shen; Mary Lou Soffa; Ricardo Baeza-Yates

**ACM U.S. Public Policy Office**  
Cameron Wilson, Director  
1100 Seventeenth St., NW, Suite 50  
Washington, DC 20036 USA  
T (202) 659-9711; F (202) 667-1066

**Computer Science Teachers Association**  
Chris Stephenson  
Executive Director  
2 Penn Plaza, Suite 701  
New York, NY 10121-0701 USA  
T (800) 401-1799; F (514) 687-1840

**Association for Computing Machinery (ACM)**  
2 Penn Plaza, Suite 701  
New York, NY 10121-0701 USA  
T (212) 869-7440; F (212) 869-0481

## STAFF

**GROUP PUBLISHER**  
Scott E. Delman  
publisher@cacm.acm.org

### Executive Editor

Diane Crawford  
**Managing Editor**  
Thomas E. Lambert  
**Senior Editor**  
Andrew Rosenblom  
**Senior Editor/News**  
Jack Rosenberger  
**Web Editor**  
David Roman  
**Editorial Assistant**  
Zarina Strakhan  
**Rights and Permissions**  
Deborah Cotton

### Art Director

Andrij Borys  
**Associate Art Director**  
Alicia Kubista  
**Assistant Art Director**  
Mia Angelica Balaquit  
**Production Manager**  
Lynn D'Addesio  
**Director of Media Sales**  
Jennifer Ruzicka  
**Marketing & Communications Manager**  
Brian Hebert  
**Public Relations Coordinator**  
Virginia Gold  
**Publications Assistant**  
Emily Eng

### Columnists

Alok Aggarwal; Phillip G. Armour;  
Martin Campbell-Kelly;  
Michael Cusumano; Peter J. Denning;  
Shane Greenstein; Mark Guzdial;  
Peter Harsha; Leah Hoffmann;  
Mari Sako; Pamela Samuelson;  
Gene Spafford; Cameron Wilson

### CONTACT POINTS

**Copyright permission**  
permissions@cacm.acm.org  
**Calendar items**  
calendar@cacm.acm.org  
**Change of address**  
acmcoa@cacm.acm.org  
**Letters to the Editor**  
letters@cacm.acm.org

### WEB SITE

http://cacm.acm.org

### AUTHOR GUIDELINES

http://cacm.acm.org/guidelines

### ADVERTISING

**ACM ADVERTISING DEPARTMENT**  
2 Penn Plaza, Suite 701, New York, NY  
10121-0701  
T (212) 869-7440  
F (212) 869-0481

### Director of Media Sales

Jennifer Ruzicka  
jen.ruzicka@hq.acm.org

### Media Kit

acmmediasales@acm.org

## EDITORIAL BOARD

**EDITOR-IN-CHIEF**  
Moshe Y. Vardi  
eic@cacm.acm.org

### NEWS

**Co-chairs**  
Marc Najork and Prabhakar Raghavan  
**Board Members**  
Brian Bershad; Hsiao-Wuen Hon;  
Mei Kobayashi; Rajeev Rastogi;  
Jeanette Wing

### VIEWPOINTS

**Co-chairs**  
Susanne E. Hambrusch;  
John Leslie King;  
J Strother Moore  
**Board Members**  
P. Anandan; William Aspray; Stefan  
Bechtold; Judith Bishop; Soumitra Dutta;  
Stuart I. Feldman; Peter Freeman;  
Seymour Goodman; Shane Greenstein;  
Mark Guzdial; Richard Heeks;  
Richard Ladner; Susan Landau;  
Carlos Jose Pereira de Lucena;  
Helen Nissenbaum; Beng Chin Ooi;  
Loren Terveen

### PRACTICE

**Chair**  
Stephen Bourne  
**Board Members**  
Eric Allman; Charles Beeler;  
David J. Brown; Bryan Cantrill;  
Terry Coatta; Mark Compton;  
Benjamin Fried; Pat Hanrahan;  
Marshall Kirk McKusick;  
George Neville-Neil

The Practice section of the ACM  
Editorial Board also serves as  
the Editorial Board of *queue*.

### CONTRIBUTED ARTICLES

**Co-chairs**  
Al Aho and Georg Gottlob  
**Board Members**  
Yannis Bakos; Gilles Brassard; Alan Bundy;  
Peter Buneman; Ghezzi Carlo;  
Andrew Chien; Anja Feldmann;  
Blake Ives; James Larus; Igor Markov;  
Gail C. Murphy; Shree Nayar; Lionel M. Ni;  
Sriram Rajamani; Jennifer Rexford;  
Marie-Christine Rousset; Avi Rubin;  
Abigail Sellen; Ron Shamir; Marc Snir;  
Larry Snyder; Manuela Veloso;  
Michael Vitale; Wolfgang Wahlster;  
Andy Chi-Chih Yao; Willy Zwaenepoel

### RESEARCH HIGHLIGHTS

**Co-chairs**  
David A. Patterson and  
Stuart J. Russell  
**Board Members**  
Martin Abadi; Stuart K. Card;  
Deborah Estrin; Shafi Goldwasser;  
Maurice Herlihy; Norm Jouppi;  
Andrew B. Kahng; Linda Petzold;  
Michael Reiter; Mendel Rosenblum;  
Ronitt Rubinfeld; David Salesin;  
Lawrence K. Saul; Guy Steele, Jr.;  
Gerhard Weikum; Alexander L. Wolf

### WEB

**Co-chairs**  
Marti Hearst and James Landay  
**Board Members**  
Jason I. Hong; Jeff Johnson;  
Greg Linden; Wendy E. MacKay



BPA Audit Pending

### ACM Copyright Notice

Copyright © 2009 by Association for Computing Machinery, Inc. (ACM).  
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to publish from permissions@acm.org or fax (212) 869-0481.

For other copying of articles that carry a code at the bottom of the first or last page or screen display, copying is permitted provided that the per-copy fee indicated in the code is paid through the Copyright Clearance Center; www.copyright.com.

### Subscriptions

Annual subscription cost is included in the society member dues of \$99.00 (for students, cost is included in \$42.00 dues); the nonmember annual subscription rate is \$100.00.

### ACM Media Advertising Policy

*Communications of the ACM* and other ACM Media publications accept advertising in both print and electronic formats. All advertising in ACM Media publications is at the discretion of ACM and is intended to provide financial support for the various activities and services for ACM members. Current Advertising Rates can be found by visiting <http://www.acm-media.org> or by contacting ACM Media Sales at (212) 626-0654.

### Single Copies

Single copies of *Communications of the ACM* are available for purchase. Please contact [acmhelp@acm.org](mailto:acmhelp@acm.org).

### COMMUNICATIONS OF THE ACM

(ISSN 0001-0782) is published monthly by ACM Media, 2 Penn Plaza, Suite 701, New York, NY 10121-0701. Periodicals postage paid at New York, NY 10001, and other mailing offices.

### POSTMASTER

Please send address changes to  
*Communications of the ACM*  
2 Penn Plaza, Suite 701  
New York, NY 10121-0701 USA



Association for  
Computing Machinery

Printed in the U.S.A.



DOI:10.1145/1516046.1516047

Elaine Weyuker

# ACM-W Celebrates Women in Computing

Computer science is no longer the hot, high-enrollment field it once was.

This is not news. While many suggestions have been made for increasing enrollments, it is unlikely that computer science will ever be as vibrant as it could be—and should be—as long as a large portion of the talent pool remains underrepresented. After all, if we are missing the best and the brightest of a group who can offer exciting ideas that would enrich the field, computer science suffers. In addition, different groups often present different perspectives—a scenario completely lost when we do not encourage diversity.

With this in mind, the mission of the ACM Women's Council (ACM-W) is to inform and support women in computing. Since ACM is an international organization, this means developing programs with a worldwide reach; with something for each of ACM's very broad constituencies: K-12 students, undergraduates at liberal arts and research institutions, master's and Ph.D. students, faculty from all types of institutions, and women in industry and government working as computer practitioners and researchers. Increasingly, we strive to partner both with other segments within ACM and other organizations dedicated to improving gender diversity.

Some of our active programs include scholarships to help women students attend research conferences. This effort is not aimed at the advanced Ph.D. student who has already committed to a career in academia or industrial research. Rather we look to support the undergraduate woman by giving her a chance to see the types of options available and encourage her to

continue on to graduate school. Similarly we hope to encourage the master's student to aim for a Ph.D. We offer up to 20 \$500 scholarships per year. Moreover, we have recently asked the ACM's Special Interest Groups (SIGs) to partner with us by offering scholarship recipients complimentary registration as well as provide conference mentors to help them learn the ropes. We are thrilled by the response we have received from many of the SIGs.

Another program involving SIG cooperation is our Athena Lecturer Award honoring the most outstanding women scholars. It was established to address the fact that women are often overlooked when nominations are considered for advanced membership grades or awards. The goal of the Athena Lecturer Award is to celebrate women's scholarship and technical contributions to the field as well as increase the visibility of women scholars. Rather than asking for individual nominations, each SIG is invited to nominate their most outstanding women scholars. By using this format, we encourage SIGs to think about promoting women in the field, and hopefully remember these women when they are nominating people for other awards or selecting keynote speakers or program chairs for future conferences.

Many readers will be familiar with the Grace Hopper Celebration of Women. To keep the Hopper momentum going throughout the year, ACM-W offers regional Hopper-like events designed to attract attendees within a two-hour driving radius of each other.

Not only does this make it relatively inexpensive to attend meetings since students and faculty often travel together, the proximity also helps establish and maintain a local community of women pursuing a common goal. We have sponsored quite a number of these meetings both within the U.S. and Australia, with one being planned in Turkey.

Another unique ACM-W initiative is the Ambassador program in which a woman serves as the Ambassador from her country and shares information about the climate there for women in computing. At times we have had representatives from six different continents. We are now developing our first internationally distributed program aimed at attracting middle school girls to computer science by adapting a successful program to several different cultures.

This is just a sampling of the many programs within ACM-W created to promote and further advance women in the computing field. Readers are encouraged to visit our Web site at <http://women.acm.org> to learn about the full range of programs and initiatives offered. ACM-W is an all-volunteer organization open to anyone interested in improving gender diversity. If you see a project that interests you, please consider volunteering. If you have an idea for a new project, let us know. Take a look at our newsletter to see project details, read interviews with outstanding women, and learn about upcoming events.

Diversity is not the problem of the underrepresented group. It is everyone's problem. If we want our field to grow and flourish, we need the contribution of talented people of all types. □

**Elaine Weyuker** is chair of ACM-W and is a researcher at AT&T Labs specializing in empirical software engineering and testing research.

© 2009 ACM 0001-0782/09/0600 \$10.00



# Congratulations

## ACM Senior Members

ACM honors 162 new inductees as Senior Members in recognition of their demonstrated performance which sets them apart from their peers

Hatim A. Aboalsamh	Leslie D. Fife	Joseph A. Leubitz	Stan Rifkin
Emad Aboelela	Stephen Fink	Pean Lim	Scott Paul Robertson
Gregory D. Abowd	Franz-Josef Fritz	Hong Lin	Ian Robinson
Divyakant Agrawal	David Garlan	Reed Little	Oscar Al. Rodriguez, II
Fadi Ahmed Aloul	Tyrone Grandison	Aurelio López-López	Nicolas F. Rouquette
Bharat B. Amberker	Rebecca Grinter	Yung-Hsiang Lu	Warren L. Rutledge
Mark Leland Ames	Ugur Gudukbay	David R. Luginbuhl	Raghvinder S. Sangwan
Abbes Amira	Mohsen Guizani	Philip Machanick	Jim Sattler
Alvaro E. Arenas	Prosenjit Gupta	Karon E. MacLean	Timoleon Sellis
Jim Ausman	David J. Haglin	Randy C. Marchany	Shubhashis Sengupta
Richard D. Austin	Chandan Haldar	Gary Marchionini	Sol M. Shatz
Turgay Aytac	Duncan Hall	Valerie A. Martin	David Singer
Felix H. Bachmann	Dieter A. Haas	José F. Martínez	Carol A. Sledge
Paolo Bellavista	Ronald C. Hart	James R. Matey	Waleed W. Smari
Michael R. Berthold	Lloyd R. Hasche	Lawrence Maturo	Cheryl L. Smith
Fernando Berzal	William E. Hefley	Kevin McCullen	Randy K. Smith
Ricardo Bianchini	Daryl H. Hepting	John F. McMullen	Oliver Spatscheck
Stefan Biffl	Douglas Hoffman	George A. Mihaila	Michael Steiner
Steven C. Bilow	Lawrence Bruce Holder	Frank W. Moore	W. Timothy Strayer
Peter Brusilovsky	Antony L. Hosking	David Moye	Xiao Su
Timothy A. Budd	Jeremy D. Impson	Carl J. Mueller	Giancarlo Succi
Fabian E. Bustamante	James Ivers	Gi-Joon Nam	Peri Tarr
Rajkumar Buyya	Daniel Á. Jiménez	Kara Nance	Alberto Tomita, Jr.
Gary J. Chastek	M. C. V. Johnson	Apostol Natsev	Gerard L. Torenvliet
Bhawani S. Chowdhry	Lawrence G. Jones	Robert L. Nord	Will Tracz
Elizabeth F. Churchill	Rosie Jones	Brian M. Novack	Elizabeth A. Unger
Terry Coatta	Srinivas Katkoori	Michael J. O'Grady	Kam Hou Vat
Sally Jo Cunningham	Edward P. Katz	Scott P. Overmyer	Tom Verhoeff
Silvester Czanner	Susan Denker Katz	Brajendra N. Panda	Stefan Voß
Meledath Damodaran	Kiyokuni Kawachiya	Priyatdarsan Patra	Yingxu Wang
Gora Datta	Michael Kearney	Ronald Perez	Jens H. Weber
Russell J. Davis	Seon Wook Kim	Birgit Pfitzmann	Bin Wei
Marios D. Dikaiakos	Andreas Kind	Claudio S. Pinhanez	Edgar R. Weippl
Sumeet Dua	Steven Kleiman	Konstantinos Psounis	Steven J. Whitehouse
Henry Been-Lirn Duh	John Krieger	Srinivasan Ramaswamy	Kent B. Wittenburg
Keith Edwards	Robert J. Kruchten	William C. Regli	David A. Wolfram
Abdulmotaleb El Saddik	Steven D. Krueger	Martin Reisslein	William Wright
L. Miguel Encarnaçao	Paul Wing Hing Kwan	Dirk Riehle	Mohamed M. Zahran
Martin Erwig	Henry M. Kyle		
Irfan Essa	Michael La Martina		
Thomas J. Essebier	Raymond Yiu Keung Lau		
Christie Ijeoma Ezeife	Raimondas Lencevicius		
Peter Farrell-Vinay	Georgios Lepouras		



Association for  
Computing Machinery

Advancing Computing as a Science & Profession

Additional ACM Senior Members will be included in an upcoming issue.

<http://seniormembers.acm.org>



# THE ACM A. M. TURING AWARD

BY THE COMMUNITY...

FROM THE COMMUNITY...

FOR THE COMMUNITY...

ACM, INTEL, AND  
GOOGLE CONGRATULATE  
**BARBARA H. LISKOV**  
FOR HER FOUNDATIONAL  
INNOVATIONS IN  
PROGRAMMING LANGUAGE  
DESIGN THAT HAVE MADE  
SOFTWARE MORE  
RELIABLE AND HER  
MANY CONTRIBUTIONS  
TO BUILDING AND  
INFLUENCING THE  
PervasivE COMPUTER  
SYSTEMS THAT POWER  
DAILY LIFE.



“ Intel is a proud sponsor of the ACM A. M. Turing Award, and is pleased to join the community in congratulating this year’s recipient, Professor Barbara Liskov. Her contributions lie at the foundation of all modern programming languages and complex distributed software. Barbara’s work consistently reflects rigorous problem formulation and sound mathematics, a potent combination she used to create lasting solutions.”

**Andrew A. Chien**  
Vice President, Corporate Technology Group  
Director, Intel Research



For more information see [www.intel.com/research](http://www.intel.com/research).

“ Google is delighted to help recognize Professor Liskov for her research contributions in the areas of data abstraction, modular architectures, and distributed computing fundamentals—areas of fundamental importance to Google. We are proud to be a sponsor of the ACM A. M. Turing Award to recognize and encourage the research that is essential not only to computer science, but to all the fields that depend on its continued advancement. ”

**Alfred Z. Spector**  
Vice President, Research and  
Special Initiatives, Google



For more information, see <http://www.google.com/corporate/index.html> and <http://research.google.com/>.

Financial support for the ACM A. M. Turing Award is provided by Intel Corporation and Google.





## Online Books & Courses Programs!

Helping Members Meet Today's Career Challenges



### NEW! Over 2,500 Online Courses in Multiple Languages Plus 1,000 Virtual Labs from Element K!



ACM's new Online Course Collection includes over **2,500 online courses in multiple languages, 1,000 virtual labs, e-reference tools, and offline capability**. Program highlights:

**The ACM E-Learning Catalog** - round-the-clock access to 2,500+ online courses on a wide range of computing and business topics, in multiple languages.

**Exclusive vLab® Virtual Labs** - 1,000 unique vLab® exercises place users on systems using real hardware and software allowing them to gain important job-related experience.

**Reference Tools** - an e-Reference Library extends technical knowledge outside of the classroom, plus online Executive Summaries and quick reference cards to answer on-the-job questions instantly.

**Offline Player** - members can access assessments and self-study courses offline, anywhere and anytime, without a live Internet connection.

A downloadable Quick Reference Guide and a 15-minute site orientation course for new users are also available to help members get started.

The ACM Online Course Program is open to ACM Professional and Student Members.

### 600 Online Books from Safari

ACM members are eligible for a **special 40% savings** offer to upgrade to a Premium or Full Library subscription through June 15, 2009.

For more details visit:

[http://pd.acm.org/books/about\\_sel.cfm](http://pd.acm.org/books/about_sel.cfm)

The ACM Online Books Collection includes **full access to 600 online books** from Safari® Books Online, featuring leading publishers including O'Reilly. Safari puts a complete IT and business e-reference library right on your desktop. Available to ACM Professional Members, Safari will help you zero in on exactly the information you need, right when you need it.



Association for Computing Machinery

*Advancing Computing as a Science & Profession*

### 500 Online Books from Books24x7

All Professional and Student Members also have **full access to 500 online books** from Books24x7®, in ACM's rotating collection of complete unabridged books on the hottest computing topics. This virtual library puts information at your fingertips. Search, bookmark, or read cover-to-cover. Your bookshelf allows for quick retrieval and bookmarks let you easily return to specific places in a book.



[pd.acm.org](http://pd.acm.org)  
[www.acm.org/join](http://www.acm.org/join)

DOI:10.1145/1516046.1516049

# Share the Threats

**O**THMAN EL MOULAT's comment "What Role for Computer Science in the War on Terror?" (Apr. 2009) concerning the article "The Topology of Dark Networks" by Jennifer Xu and Hsichun Chen (Oct. 2008) that the views and articles in *Communications* should have no bearing on or bias toward any agenda, political or religious, is a point well taken. However, in light of the security breaches occurring throughout the digital world, any information that exposes threats should indeed be well received and published wherever it is relevant to technologists and security specialists, as in *Communications*.

It is reasonable to suspect that potential terrorist cells or factions willingly and wantonly seek ways to destroy Western technologies and organizations. An article aimed at exposing threats or educating the public on future threats does not in any way target a specific race, creed, or religion.

I applaud the authors of "The Topology of Dark Networks" and hope *Communications* continues to keep us up to date with factual articles of this nature. Organizations that are concerned with their own beliefs, traditions, and objectives should be willing to transparently share their interests with the rest of the world.

**John Orlock, IL**

## Virtualization Still Evolving

Kirk L. Kroeker's news article "The Evolution of Virtualization" (Mar. 2009) took a limited view of its subject. I contrast it with how my company uses virtual machines for several quite practical purposes:

**Software testing.** Rather than build a test environment, then rebuild it after a series of tests, we set it up with a set of baseline virtual machines (perhaps client/server), then run our tests. This way when we finish testing, we copy back over the baseline virtual machine and are ready for the next round of testing;

**Customer support.** We look to mimic customer configurations in a set of vir-

tual machines, aiming to verify or refute a problem a customer might be having and possibly provide a workaround or new build of the software. If this scenario turns out to be common, we roll it into our testing sandbox;

**Project services.** When trying to remotely configure and build a solution for a customer, we first build it in a virtual machine, then apply the solution and test. This process also greatly improves delivery of the solution; and

**Software demonstration.** We build our demos in a virtual machine, making it much easier for us to get them out to field personnel.

**Jerry Walter, Troy, OH**

## How to Define the Granularity of Properties and Functions

I was confused about the discussion of properties and functions in Daniel Jackson's review article "A Direct Path to Dependable Software" (Apr. 2009). Jackson seemed to be saying that properties are more fine-grain than functions yet also that a property cuts across several functions at the same time. Doesn't this imply that properties are coarse-grain, assuming they transcend several functions?

Trying to resolve my questions with the help of Webster's dictionary, I learned that a function is a "factor" and a property is any attribute or characteristic. So functions and properties can be both fine- and coarse-grain, depending on the assumptions of abstraction inherent in the mind of the author.

Does Jackson view a "function" as a modularity construct in a programming language? Does he mean that properties are those factors or attributes ("functions" if you will) that are independent of the software's special-case implementation?

I may still be confused, but trying to infer Jackson's meaning led me to conclude the following: Fine-grain attention to the software's behavior-level characteristics (including properties, functions, or whatever abstractions a developer is using) is important in de-

fining an effective dependability case.

Is this correct?

**CJ Fearnley, Upper Darby, PA**

## Author's Response:

*Requirements traditionally break the behavior of a system into a collection of functions, each describing in full some feature of the system. A radiotherapy machine might, for example, offer functions to recall a patient's prescribed dose from a database; set the equipment to deliver a given dose; activate the equipment; and so on. Prioritizing functions isn't very useful, because the critical aspects of a system typically involve many functions, though often not in their entirety.*

*A property, on the other hand, describes an expected observation of the system's behavior and can be expressed at any level of granularity: that, for example, some of the dose delivered to a patient never exceeds some fixed limit; that a patient receives his or her prescribed dose within some tolerance; that the dose delivered and the dose logged always match; and so on. So a property can at the same time be more fine-grain than a single function (since it describes the function only partially) and cut across multiple functions.*

**Daniel Jackson, Cambridge, MA**

## Corrections

In the Q&A "Our Dame Commander" (Apr. 2009), Leah Hoffmann described Wendy Hall as "the third female president of ACM." Hall is the sixth, preceded by: Jean Sammet (1974–1976), Adele Goldberg (1984–1986), Gwen Bell (1992–1994), Barbara Simons (1998–2000), and Maria Klawe (2002–2004).

The photographs of the Rebooting Computing Summit (Apr. 2009) were taken by Richard P. Gabriel (page 2) and by Mary Bronzan (page 19).

*Communications* welcomes your opinion. To submit a Letter to the Editor, please limit your comments to 500 words or less and send to letters@cacm.acm.org.

© 2009 ACM 0001-0782/09/0600 \$10.00



The *Communications* Web site, [cacm.acm.org](http://cacm.acm.org), features 13 bloggers in the BLOG@CACM community. In each issue of *Communications*, we'll publish excerpts from selected posts, plus readers' comments.

DOI:10.1145/1516046.1516072

[cacm.acm.org/blogs/blog-cacm](http://cacm.acm.org/blogs/blog-cacm)

# Speech-Activated User Interfaces and Climbing Mt. Exascale

Tessa Lau discusses why she doesn't use the touch screen on her in-car GPS unit anymore and Daniel Reed considers the future of exascale computing.



## From Tessa Lau's "Hello, Computer"

Four years ago when I bought my first in-car Global Positioning System (GPS) unit, it felt like a taste of the future. The unit knew where I was, and regardless of how many wrong turns I made, it could tell me how to get where I wanted to go. It was the ultimate adaptive interface: No matter where I started, it created a customized route that would lead me to my destination.

Alas, my first GPS unit met an untimely end in a theft involving a dark night, an empty street, and a smashed window.

My new GPS, a Garmin nüvi 850, comes with a cool new feature: speech-activated controls.

Speech recognition brings a new dimension to the in-car human-computer interface. When you're driving, you're effectively partially blind and have no hands. Being able to talk to the computer and instruct it using nothing but your voice is amazingly

empowering, and makes me excited about the future of voice-based interfaces.

The nüvi's interface is simple and well designed. There's a wireless, button-activated microphone that you mount to your steering wheel. When you activate the mic, a little icon appears on the GPS screen to indicate that it's listening, and the GPS plays a short "I'm listening" tone. You can speak the names of any buttons that appear on the screen or one of the always-active global commands (e.g., "main menu," "music player," or "go home"). Musical tones indicate whether the GPS has successfully interpreted your utterance. If it recognized your command, it takes you to the next screen and verbally prompts you for the next piece of information (e.g., the street address of your destination). Most of the common GPS functionality can be activated via spoken confirmations without even looking at the screen.

Lists (e.g., of restaurant names) are annotated with numbers so you

only have to speak the number of the item you want from the list. However, it also seems to correctly recognize the spoken version of anything in the list, even if it's not displayed on the current screen (e.g., the name of an artist in the music player).

In my tests it's been surprisingly accurate at interpreting my speech, despite the generally noisy environment on the road.

What has surprised me the most about this interface is that the voice-based control is so enjoyable and fast that I don't use the touch screen anymore. Speech recognition, which had been in the realm of artificial intelligence for decades, has finally matured to the point where it's now reliable enough for use in consumer devices.

Part of the power of the speech-activated user interface comes from the ability to jump around in the interface by spoken word. Instead of having to navigate through several different screens by clicking buttons, you can jump straight to the desired screen by speaking its name. It's reminiscent of the difference between graphic user interfaces (GUIs) and command lines; GUIs are easier to learn, but once you master them, command lines offer more efficiency and power. As is the case with command lines, it takes some experimentation to discover what commands are available when; I'm still learning about my GPS and how to control it more effectively.

Kudos, Garmin, you've done a great

job with the nüvi 850. I can't wait to see what the future will bring! (Voice-based access to email on the road? It seems almost within reach.)

*Disclaimer: The views expressed here do not necessarily reflect the views of my employer, ACM, or any other entity besides myself.*

#### Reader's comment:

*Information I've read lately on the topic of speech recognition indicates that a device's ability to correctly recognize commands depends in large measure on the quietness of the environment. I have often found that voice systems on my cell phone don't work well unless I find a quiet place to access them. So it is good to hear that Garmin has found an effective way to interpret commands while driving—an environment that you note can be noisy.*

*As you speak of future enhancements, it brings up the issue of what drivers should be able to do while on the road. Multitasking is great, but I'm not sure email while driving is such a good idea...*

—Debra Gouchy



#### From Daniel Reed's "When Petascale Is Just Too Slow"

It seems as if it were just yesterday when I was at the National Center for Supercomputing Applications and we deployed a one teraflop Linux cluster as a national resource. We were as excited as proud parents by the configuration: 512 dual processor nodes (1 GHz Intel Pentium III processors), a Myrinet interconnect, and (gasp) a stunning 5 terabytes of RAID storage. It achieved a then-astonishing 594 gigaflops on the High-Performance LINPACK benchmark, and was ranked 41st on the Top500 list.

The world has changed since then. We hit the microprocessor power (and clock rate) wall, birthing the multicore era; vector processing returned incognito, renamed as graphical processing units (GPUs); terabyte disks are available for a pittance at your favorite consumer electronics store; and the top-ranked system on the Top500 list broke the petaflop barrier last year, built from a combination of multicore processors and gaming engines. The last is interest-

ing for several reasons, both sociological and technological.

#### Petascale Retrospective

On the sociological front, I remember participating in the first peta-scale workshop at Caltech in the 1990s. Seymour Cray, Burton Smith, and others were debating future petascale hardware and architectures, a second group was debating device technologies, a third was discussing application futures, and a final group of us was down the hall debating future software architectures. All this was prelude to an extended series of architecture, system software, programming models, algorithms, and applications workshops that spanned several years and multiple retreats.

At the time, most of us were convinced that achieving petascale performance within a decade would require new architectural approaches and custom designs, along with radically new system software and programming tools. We were wrong, or at least so it superficially seems. We broke the petascale barrier in 2008, using commodity x86 microprocessors and GPUs, InfiniBand interconnects, minimally modified Linux, and the same message-based programming model we have been using for the past 20 years.

However, as peak system performance has risen, the number of users has declined. Programming massively parallel systems is not easy, and even terascale computing is not routine. Horst Simon explained this with an interesting analogy, which I have taken the liberty of elaborating slightly. The ascent of Mt. Everest by Edmund Hillary and Tenzing Norgay in 1953 was heroic. Today, amateurs still die each year attempting to replicate the feat. We may have scaled Mt. Petascale, but we are far from making it pleasant or even a routine weekend hike.

This raises the real question: Were we wrong in believing different hardware and software approaches would be needed to make petascale computing a reality? I think we were absolutely right that new approaches were needed. However, our recommendations for a new research and development agenda were not realized. At least, in part, I believe this is because

we have been loathe to mount the integrated research and development needed to change our current hardware/software ecosystem and procurement models.

#### Exascale Futures

Evolution or revolution, it's the persistent question. Can we build reliable exascale systems from extrapolations of current technology or will new approaches be required? There is no definitive answer as almost any approach might be made to work at some level with enough heroic effort. The bigger question is: What design would enable the most breakthrough scientific research in a reliable and cost-effective way?

My personal opinion is that we need to rethink some of our dearly held beliefs and take a different approach. The degree of parallelism required at exascale, even with future many-core designs, will challenge even our most heroic application developers, and the number of components will raise new reliability and resilience challenges. Then there are interesting questions about many-core memory bandwidth, achievable system bisection bandwidth, and I/O capability and capacity. There are just a few programmability issues as well!

I believe it is time for us to move from our deus ex machina model of explicitly managed resources to a fully distributed, asynchronous model that embraces component failure as a standard occurrence. To draw a biological analogy, we must reason about systemic organism health and behavior rather than cellular signaling and death, and not allow cell death (component failure) to trigger organism death (system failure). Such a shift in world view has profound implications for how we structure the future of international high-performance computing research, academic/government/industrial collaborations, and system procurements. □

Tessa Lau is a research staff member at IBM Almaden Research Center in San Jose, CA. Daniel Reed is director of scalable and multicore systems at Microsoft Research in Redmond, WA.

© 2009 ACM 0001-0782/09/0500 \$10.00



DOI:10.1145/1516046.1516050

David Roman

## Making that Connection

The goal of holding readers' attention has made provocation a timeworn editorial strategy. *Communications* doesn't resort to screaming headlines like most storefront fare, but it does strive to publish eye-catching imagery for its must-read articles. This month's cover story, "One Laptop Per Child: Vision vs. Reality," with its title's inherent tension, is a case in point.

*Communications* also aims for authority; its articles can be a beginning as much as an end. The "Viewpoints" pages, for example, may introduce unsettled and unsettling ideas that prompt readers to react and respond not only to the editorial but to each other. Indeed, the recent debate on network neutrality that was first presented in the pages of the February issue, continued into the May issue, and it's hardly over yet.

You can be a part of this debate at [cacm.acm.org](http://cacm.acm.org). *Communications'* Web site invites and lends itself to quick feedback via the "User Comments" feature that allows a continued conversation about a topic. Reachable from the "Tools for Readers" at the top right of each article page, and at the bottom of every article page, the feature requires a simple sign-in (so we can follow who's speaking). From there, readers are welcome to present what Editor-in-Chief Moshe Vardi calls "well-reasoned and well-argued opinions" to keep the discussion lively. I encourage all readers to start or join an online discussion.

### Wanted: Expert Bloggers

Ever consider yourself a blogger? If so, we should talk.

*Communications* wants to expand its ever-evolving roster of expert bloggers. Experience is a plus but credentials and passion are equally important. The level of commitment we require is open-ended; if you are willing to work with us, we will accommodate your schedule. If you are interested but cannot add it to your workload at the moment, we could put you on our future schedule or at least get you on our radar.

In addition, if you follow the blogs of someone you consider a good fit for *Communications*, we'd like to hear your recommendations. Contact us at [blog@cacm.acm.org](mailto:blog@cacm.acm.org).



## ACM Member News



### EMER WINS ECKER-MAUCHLY AWARD

ACM and the IEEE Computer Society will jointly present the Eckert-Mauchly Award to Joel Emer, director of microarchitecture research at Intel, for pioneering contributions to performance analysis, modeling methodologies, and design innovations in several significant industry microprocessors. Emer developed quantitative methods including measurement of real machines, analytical modeling, and simulation techniques that are now widely employed to evaluate the performance of complex computer processors. Emer will receive the 2009 Eckert-Mauchly Award, the most prestigious award in the computer architecture community, at the International Symposium on Computer Architecture, June 20–24, in Austin, TX.

### EGGERS RECEIVES ATHENA LECTURER AWARD

Susan Eggers, a professor of computer science and engineering at the University of Washington, has won ACM's 2009–2010 Athena Lecturer Award. Eggers' work on computer architecture and experimental performance analysis led to the development of Simultaneous Multithreading, the first commercially viable multithreaded architecture. This technique improves the overall efficiency of certain processors known as superscalar and has been adopted by Intel, IBM, and others.

### WHITNEY RECOGNIZED FOR DISTINGUISHED SERVICE

ACM presented the Distinguished Service Award to Telle Whitney for her profound impact on the participation of women in computing. Whitney, president and CEO of the Anita Borg Institute for Women and Technology, cofounded the Grace Hopper Celebration of Women in Computing, which has grown into an annual event. The conference is widely recognized as one of the best ways to encourage women to major in computing, continue on to graduate school, and pursue a career in computing.

Science | DOI:10.1145/1516046.1516051

Don Monroe

## Micromedicine to the Rescue

*Medical researchers have long dreamed of “magic bullets” that go directly where they are needed. With micromedicine, this dream could become a life-saving reality.*

A HEADACHE or other pain will send many of us to the medicine cabinet for a pain reliever. Molecules from the swallowed pill quickly find their way directly to the source of the pain. But how do they know where to go? Of course, they don’t; the molecules travel throughout the body, chemically reacting wherever they can.

The consequences of “broadcasting” drugs to the whole body are profound. Drugs that attack rogue, cancer-causing cells also afflict other dividing cells, such as those in the intestine. In fact, chemotherapy doses are often reduced to avoid nausea and other unpleasant side effects, and other, more powerful drugs are too toxic to even be considered.

Researchers have long dreamed of “magic bullets” that go directly where they are needed. Indeed, many current drugs are formulated to be taken up by particular tissues, and nanotechnology is giving researchers even more delivery options. But what if the delivery system could “diagnose” the local conditions? In contrast to today’s “dumb envelopes,” Ehud Shapiro, a computer

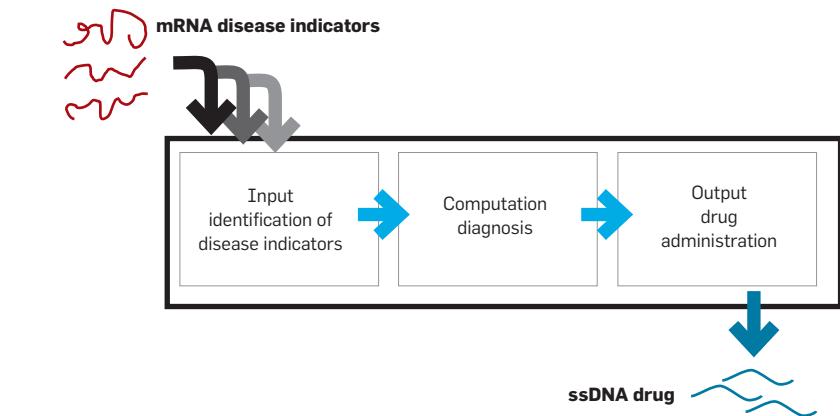
scientist and biological chemist at the Weizmann Institute of Science in Rehovot, Israel, likens this approach to a “smart envelope.” The envelope “would open up only at the right place and the right time for the specific action,” such as releasing a potent but toxic cancer drug, he says. “This would open up a whole range of molecules that are totally inaccessible today as drugs.”

In addition to delivering drugs, microscopic agents could transform the regeneration of damaged tissues and the diagnosis of disease. The time until

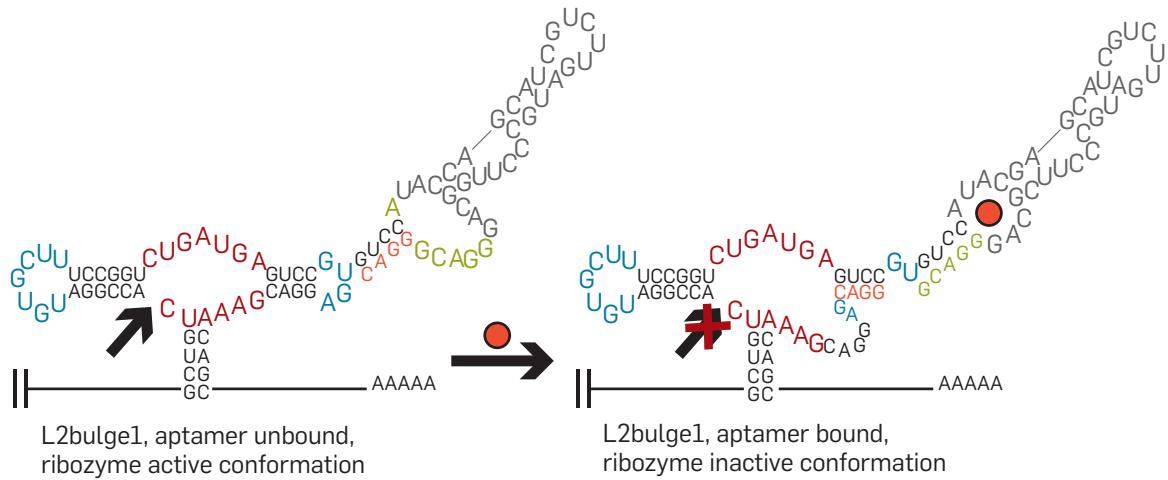
these ideas help patients is “probably measured in decades, not in years,” Shapiro admits. Long before that, however, researchers could use the new tools to explore biology in the lab. The challenge of engineering biology, rather than merely observing it, could yield powerful insights into how biological systems work.

### Hijacking Biology

Recent years have been revolutionary for biology. The human genome, as well as computer-based tools that measure thousands of biological chemicals simultaneously, have inundated biologists with data about how these chemicals interact to create the processes of life. An eager group of researchers around the world take this data glut as a challenge to build new biological circuits from scratch, in what is known as “synthetic biology.” Using various strategies, they are assembling pieces



An example of a test-tube “molecular computer” created by Ehud Shapiro and colleagues.



**An example of a synthetic riboswitch engineered by Maung Nyan Win and Christina Smolke in which the ribozyme is turned off when the aptamer binds ligand.**

that might enable completely new approaches to medical technology.

In 2004, for example, Shapiro and his colleagues created a test-tube “molecular computer” consisting of three interconnected modules. The first module sensed the concentrations of four types of messenger RNA, the working copies of the genetic instructions in DNA, which are used to produce proteins. The second module performed a “diagnosis,” computing whether two of the messenger RNA levels decreased while two others increased, a signature that might indicate a disease. Depending on the results of the computation, the third module dispensed a drug molecule. “We demonstrated the whole process, beginning to end,” Shapiro notes, “but in a test tube.”

To both sense specific strands of messenger RNA and to perform the computation, the Weizmann researchers exploited the sequence-specific matching of DNA strands. So far, though, they have not operated their molecular computer in the complex environment of a living cell. Other teams have had success with different schemes. For instance, a group including Shapiro’s former collaborator Yaakov Benenson, now a researcher at the EAS Center for Systems Biology at Harvard University, demonstrated computation—but not sensing—in cultured human kidney cells. They exploited the newly discovered phenomenon of RNA interference, in which the presence of short RNA templates activates cellular mechanisms that suppress protein synthesis for matching messenger RNA.

A Caltech team led by Christina Smolke, now a professor of bioengineering at Stanford University, designed complex RNA molecules that included three separate sections, performing sensing, computation, and actuation. Although all three modules are part of the same molecule, they act independently, so the function of each part can be separately modified, she says. “You have this plug-and-play type capability to build many types of functions from a smaller set of modular components.” The RNA molecules they design are manufactured by yeast or even mammal cells after the researchers insert the corresponding DNA.

In addition to computing Boolean logic operations, Smolke’s team has demonstrated other signal-processing functions, including bandpass filtering and adjustable signal gain, with their RNA platform. But she acknowledges

that the field has yet to settle on the best approach. “Ultimately, you want to get to a place where there’s some level of standardization,” she says.

### Send in the Clones

One barrier to standardization is the wide range of possibilities for using biological agents in medicine. Smolke’s technique, for example, might be used to genetically modify cells in a particular tissue, but she is also exploring modification of immune cells outside of the body to combat cancer. “We’re utilizing the function that [the immune cell] already does really well, and then endowing it with enhanced functions,” she says.

Chris Anderson, a professor in the department of bioengineering at the University of California, Berkeley, envisions a different strategy, one based on engineered bacteria, but admits that “it’s impossible to know what’s going to win out.” A “huge advantage” of using bacteria, Anderson says, is that the biological processes targeted by anti-bacterial agents are very different from those of human cells, so the engineered bacteria can be easily killed.

For bacteria to be effective, they must be able to evade the body’s immune defenses. Anderson and his colleagues have transplanted genes from other bacteria that allow their *E. coli* to survive for hours in the bloodstream, instead of just a few minutes. They also introduced growth-control mechanisms into the bacteria, he stressed. “They’re not able to grow without feeding something to the patient.”

**In addition to computing Boolean logic operations, a Caltech team has demonstrated other signal-processing functions with its RNA platform.**

In 2006, Anderson and his colleagues unveiled a bacterium they had engineered to invade nearby cells. Importantly, the invasion only occurred under chosen conditions, including lack of oxygen, which often occurs near tumors. Rather than directly combining sensing, computation, and actuation into a single DNA or RNA molecule, Anderson's genetic modules communicate using smaller molecules, in much the same way as normal cells. When the researchers insert new DNA into the bacteria, they include special sequences that respond to other chemicals in the cell or the environment. They "connect" their modules by inducing this sensitivity to the products of other genes that they insert. In addition, by requiring that two different molecules attach to adjacent regions of DNA, they created the cellular equivalent of an AND gate.

### A Need to Communicate

The chemical sensitivity of genes gives cells some ability to communicate with each other. For example, one of the signals that stimulated Anderson's bacteria to invade was the well-known "quorum-sensing" response that kicks in for some bacteria when they are present in large numbers. Ron Weiss, a professor of electrical engineering and molecular biology at Princeton University, has used the quorum-sensing machinery to build bidirectional communication between two groups of bacteria. The collective behavior constitutes a kind

of computation that reflects the interaction between the two strains, each of which could be tuned to detect separate conditions. In some cases, Weiss says, "a cell that specializes in the detection of one condition can do it much better than a cell that tries to do too many things at once."

From a broader perspective, says computer scientist Tatsuya Suda of the University of California at Irvine, "there's always communication involved" in micromedicine. The sensing of the environment by the tiny agents is a kind of communication, he notes, as is the dispensing of drugs. As researchers design these tiny communications systems, he stresses, they need to pay careful attention to noise.

In addition to communicating with their environment, microscopic agents may communicate with each other. As an example, Suda cites regenerative medicine, in which the creation of a replacement organ requires coordinated response by many agents. But he admits that, for now, "the state of the art is just trying to find out how they work together as a group, as opposed to how we can take advantage of group behavior."

For biologically based agents, as for ordinary cells, any communication is likely to occur through the emission and sensing of molecules. In contrast, artificial or hybrid systems incorporating nanometer-scale electronic components might also communicate by ultrasound or radio. In principle, as

described by researcher Tad Hogg of Hewlett-Packard Labs, they could signal to point others to medically important locations. In addition, they might be able to transmit information to the outside world.

Augmenting, rather than replacing, the diagnostic strengths of the medical community could be an important early application of micromedicine, and relaxes the demands for on-board computation and drug delivery. At a minimum, small devices might extend the capabilities of chemicals whose locations are monitored in modern medical equipment. "As those imaging devices advance," Hogg says, "they should be able to give you some information more than just 'here' or 'not here,' but what they found" in a particular region, perhaps by combining several important local measurements.

Even before medical applications become practical, Shapiro suggests, the emerging tools could provide new resources for basic biology research. "I think that these types of molecular computing devices might be able to analyze living cells *ex vivo* and help researchers understand cells without killing them," Shapiro notes. "These applications are probably measured in years rather than in decades." □

**Don Monroe** is a science and technology writer based in Murray Hill, NJ.

© 2009 ACM 0001-0782/09/0600 \$10.00

### Search Technology

## Kleinberg Wins ACM-Infosys Foundation Award

Jon Kleinberg, a professor of computer science at Cornell University, is the winner of the 2008 ACM-Infosys Foundation Award in the Computing Sciences for his contributions to improving Web search techniques that allow billions of Web users worldwide to find relevant, credible information on the ever-evolving Internet. Kleinberg, 37, developed models that document how information is organized on the Web, how it spreads through large social networks, and how these networks are structured to create the small-world

phenomenon known as "six degrees of separation."

Kleinberg's use of mathematical models to illuminate search and social networking tools that underpin today's social structure has created interest in computing from people not formerly drawn to this field. The ACM-Infosys Foundation Award, established in 2007, recognizes personal contributions by young scientists and system developers to a contemporary innovation that exemplifies the greatest recent achievements in the computing field. Financial

support for the \$150,000 award is provided by an endowment from the Infosys Foundation.

"Professor Kleinberg's achievements mark him as a founder and leader of social network analysis in computer science," says Professor Dame Wendy Hall, president of ACM. "With his innovative models and algorithms, he has broadened the scope of computer science to extend its influence to the burgeoning world of the Web and the social connections it enables. We are fortunate to have the benefit of his profound

insights into the link between computer network structure and information that has transformed the way information is retrieved and shared online."

The ACM-Infosys Foundation Award recognizes young researchers who are currently making sizeable contributions to their fields and furthering computer science innovation. The goal is to identify scientifically sound breakthrough research with potentially broad implications, and encourage the recipients to further their research.

# Content Control

*Entertainment businesses say digital rights management prevents the theft of their products, but access control technologies have been a uniform failure when it comes to preventing piracy. Fortunately, change is on the way.*

**B**Y NOW, THE story is familiar: CD sales are falling. Digital music sales are growing, but have not offset the loss. The music business is struggling to adapt to a new technological era. It's not the first time. At the turn of the 20th century, for instance, as the phonograph gained popularity, the industry's model of compensation and copyright was suddenly thrown

into question. Previously, composers of popular songs relied on the sale of sheet music for their income. After all, musicians needed sheet music to learn and perform a work, even if individual performances generated no royalties. Once performances could be recorded and sold or broadcast on the radio, however, the system grew less appealing to both groups of artists, who were essentially getting paid once

for something that could be consumed thousands of discrete, different times. Eventually, collection societies were set up to make sure each party had a share in the new revenue streams.

Today, musical copyright is most prominently embodied not by sheet music but by audio recordings, along with their translations and derivatives (that is, their copies). Yet computers have made light work of reproducing most audio recordings, and the industry is unable to prevent what many young fans are now used to—free copies of their favorite songs from online file-sharing networks like BitTorrent and LimeWire. Legal barriers, like the injunctions imposed by the Digital Millennium Copyright Act (DMCA) against copying protected works or circumventing their digital protections, are unpopular and difficult to enforce. (The industry's John Doe suits have touched a mere fraction of file sharers, and their effect on the overall volume of illegal downloads is questionable.) Technological barriers, like the widespread security standards and controls known as digital rights management (DRM), have been even less effective.

DRM attempts to control the way digital media are used by preventing purchasers from copying or converting them to other formats. In theory, it gives content providers absolute power over how their work is consumed, enabling them to restrict even uses that are ordinarily covered by the fair use doctrine. Purchase a DVD in Europe, and you'll be unable to play it on a DVD player in the U.S. because of region-coding DRM. What's more, according to the DMCA, it would be illegal for you to copy your DVD's contents into a different format, or otherwise attempt to circumvent its region-coding controls. To take a musical example, until recently songs purchased in Apple's



By putting copyrighted books online, Google Book Search may soon revolutionize book publishing.

popular iTunes music store could only be played on an iPod due to the company's proprietary DRM.

Entertainment businesses say they need DRM to prevent the theft of products that represent their livelihood. In practice, however, DRM has been a uniform failure when it comes to preventing piracy. Those who are engaged in large-scale, unauthorized commercial duplication find DRM "trivial to defeat," says Jessica Litman, a professor of law at the University of Michigan. The people who don't find it trivial: ordinary consumers, who are often frustrated to discover that their purchases are restricted in unintuitive and cumbersome ways.

In the music industry, at least, change is underway. In 2007, Amazon announced the creation of a digital music store that offered DRM-free songs, and in January 2009, Apple finalized a deal with music companies to remove anti-copying restrictions on the songs it sold through iTunes. Since iTunes is the world's most popular digital music vendor—and the iPod its most popular player—critics complained the deal would only further solidify Apple's hold on the industry. Yet because consumers can now switch to a different music player without losing the songs they've purchased, the prediction seems dubious.

"As long as the cost of switching technologies is low, I don't think Apple will exert an undue influence on consumers," says Edward Felten, a professor of computer science and public affairs at Princeton University.

What about piracy? Since DRM never halted musical piracy in the first place, experts say, there's little reason to believe its absence will have much effect. In fact, piracy may well decrease thanks to a tiered pricing scheme in the Apple deal whereby older and less popular songs are less expensive than the latest hits. "The easier it is to buy legitimate high-quality, high-value products," explains Felten, "the less of a market there is for pirated versions." By way of illustration, he points to the 2008 release of *Spore*, a hotly anticipated game whose restrictive DRM system not only prevented purchasers from installing it on more than three computers, but surreptitiously installed a separate program called SecuROM

## DRM is being "wielded as a powerful tool" against unapproved technologies, says Aaron Perzanowski.

on their hard drives. Angry gamers responded by posting copies of the game online, making *Spore* the most pirated game on the Internet.

### DRM and Movies

Yet DRM is nowhere near dead outside the music business. Hollywood, protected thus far from piracy by the large file size of the average feature film, continues to employ it as movies become available through illegal file-sharing networks. Buy a movie on iTunes, and you'll still face daunting restrictions about the number and kind of devices you can play it on. Buy a DVD, and you'll be unable to make a personal-use copy to watch on your laptop or in the car.

DRM has also proven useful as a legal weapon. Kaleidescape, a company whose digital "jukeboxes" organize and store personal media collections, was sued in 2004 by the DVD Content Control Association, which licenses the Content Scrambling System that protects most DVDs. (In 2007, a judge ruled there was no breach of the license; the case is still open on appeal.)

The Kaleidescape case is instructive, experts say, since it shows that preventing piracy isn't necessarily Hollywood's biggest concern. Entry-level Kaleidescape systems start at \$10,000—unlikely purchases for would-be copyright infringers. "Instead, DRM is wielded as a powerful tool to prevent the development and emergence of unapproved technologies. In some instances, that may overlap with some concern over infringement, but as the Kaleidescape example shows, it need not," says Aaron Perzanowski, a research fellow at the Berkeley Center for Law & Technology.

Indeed, the real question typically comes down to one of business mod-

els: Can companies preserve their current revenue structures through DRM or in court, or must they find some other way of making money? For music, the iTunes model appears to be a viable one, though questions still remain. For movies, the path is less clear. What will happen when DVDs become obsolete? Will consumers take out subscriptions to online movie services, or make discrete one-time purchases? "Nobody knows what the marketplace of the future will look like," says Litman. And the wholesale copyright reform that digital activists long for is years away.

One industry whose business model may soon be radically transformed is publishing. Under the terms of a recent settlement reached with the Authors Guild (which sued Google in 2005 to prevent the digitization and online excerpting of copyrighted books as part of its Book Search project), Google agreed to set up a book rights registry to collect and distribute payments to authors and publishers. Much like the collection societies that were established for musicians, the registry would pay copyright holders whenever Internet users elected to view or purchase a digital book; 63% of the fee would go to authors and publishers, and 37% to Google.

If approved, the settlement would be "striking in its scope and potential future impact," says Deirdre Mulligan, a professor of law at the University of California, Berkeley's School of Information. It is nonetheless highly controversial. Some, like James Grimmelmann, a New York Law School professor, believe it is a "universal win compared with the status quo." Others are disappointed by what they see as a missed opportunity to set a powerful court precedent for fair use in the digital age, and the undeniable danger of monopoly. "No other competitors appear poised to undertake similar efforts and risk copyright legislation," says Perzanowski.

One thing, at least, is clear: It frees the courts to consider other industries' complaints as they slouch toward the digital age. □

**Leah Hoffmann** is a Brooklyn, NY-based science and technology writer.

# Autonomous Helicopters

*Researchers are improving unmanned helicopters' capabilities to address regulatory requirements and commercial uses.*

HERE WOULD SEEM to be a clear market niche for unmanned helicopters. Equipped with lightweight onboard cameras, they could serve as mapping agents or search-and-rescue "eyes" in places where using a full-sized helicopter and a human crew are life threatening or cost prohibitive. Motion-picture producers have explored the use of autonomous helicopters in filming action scenes in locations where the safety of both flight crews and movie cast members could be at risk from using larger aircraft. Humanitarian groups have considered using autonomous helicopters for land-mine detection, while public safety agencies have explored using them for inspecting bridges and other structures where human inspectors might be endangered. And they are becoming mainstays in applications such as crop dusting in Japan, where the need to fly at a low altitude and spray chemicals can be dangerous for pilots.

Academic and commercial research teams have been perfecting the capabilities of autonomous helicopters for nearly two decades, with such widespread deployments as a goal. Algorithmic and technological advances are occurring at a steady pace, but regulatory roadblocks and trade restrictions are hampering market acceptance. And, though much of the cutting-edge research in autonomous helicopters demonstrates significant crossover potential between disparate computational and scientific disciplines as well as other aviation applications, many researchers find themselves stymied by these non-technological obstacles that stem from policy concerns.

"A lot of vehicles have at least kinematics that are similar to helicopters," says Adam Coates, a Stanford University Ph.D. student who coauthored *Learning for Control from Multiple Demonstrations*, which won the



One of Stanford University's autonomous helicopters flying upside down in an aerobically challenging airshow. For more photos and video, visit <http://heli.stanford.edu/index.html>.

International Conference for Machine Learning's best application paper for 2008, and describes how he and colleagues programmed an autonomous helicopter to perform complex aerobatics. "But I think the biggest hurdle is regulatory. It's virtually impossible to do real UAV [unmanned aerial vehicle] operations unless you're a defense contractor or the military—so you have to go to a big defense contractor if you want to do real UAV research."

Regulatory hurdles vary, depending on the sovereignty involved. In the U.S., for example, the Federal Aviation Administration (FAA) has yet to issue regulations regarding the use of autonomous helicopters in public airspace. A 2008 report by the U.S. General Accountability Office (GAO) noted that unmanned aircraft, whether fixed wing or rotor powered, cannot meet the National Airspace System's safety regulations for tasks such as avoiding

other aircraft. Therefore, autonomous crafts' use is limited to case-by-case approval by the FAA, and usually restricted to line-of-sight operation. In Japan, the government has placed strict trade restrictions on the Yamaha RMAX autonomous helicopter, which is regarded as the industry benchmark, to prevent it from being used for military operations by unfriendly nations.

Omead Amidi, a research faculty member at Carnegie Mellon University and CEO of SkEyes Unlimited, a Washington, PA-based firm that manufactures instruments for autonomous aircraft, concurs with Coates' observation about the dearth of regulatory infrastructure hindering wider development and deployment of the craft.

"If you have a helicopter flying over your head, it's because everything about it is regulated," Amidi says. "No such thing exists for autonomous helicopters. If you could convince me to

fly one of these over the head of my daughter, OK, it's ready, but I'm not doing it now."

#### **AI to the Forefront**

Despite the regulatory issues, which the GAO estimated might take 10 years to resolve in the U.S., researchers have continued to improve autonomous helicopters' capabilities. The most advanced can take off, hover, and maintain flight autonomously through a combination of advanced sensing and navigation equipment such as laser sensors, GPS modules, inertial measurement units that contain accelerometers and gyroscopes, and communications modules that communicate with ground-based computers or human pilots when necessary. The RMAX, for example, first flew fully autonomously out of visual range in Japan in 2000, following preprogrammed instructions.

While the RMAX is well suited for commercial purposes, it is also prohibitively large and expensive for applications such as the surveillance of building interiors or for bootstrapped university research programs. A base model used by the U.S. Army for research weighs approximately 185 pounds, has a rotor diameter of three meters, and costs \$86,000, while fully autonomous units, complete with navigational and control equipment, can cost \$1 million.

Researchers are successfully applying disparate technologies to improve the vehicles, using much smaller and cheaper helicopters than the RMAX. For example, Coates and coauthor Peter Abbeel, now a professor in the department of electrical engineering and computer sciences at the University of California, Berkeley, utilized artificial intelligence principles to demonstrate their assertion that an off-the-shelf expectation-maximization algorithm could result in the most advanced autonomous aerobatics yet performed, using a commercially available radio-controlled hobbyist helicopter that weighed about 10 pounds.

Coates says the Stanford project was the culmination of five years of effort, in which numerous approaches were discussed and dismissed. Andrew Ng, a professor of computer science at Stanford, who advised Coates and Abbeel

**Human-generated mapping can cost \$20,000 per square mile; an autonomous helicopter could produce the same results 10 times cheaper, says Omead Amidi.**

in their project and was a coauthor of the *Learning for Control* paper, says the project successfully transferred machine learning techniques into a discipline that had hitherto been extremely labor-intensive, relying on painstaking expert modeling of likely behaviors. Ultimately, they decided to have the helicopter "watch" an expert human pilot's maneuvers via data input from onboard controls and a radio receiver that saved a copy of the human pilot's control stick positions during demonstration flights.

"From those two things, you can examine state changes over time and what the pilot does, and can record a whole trajectory to build up a model," Coates says.

"Previously, the most common approach to designing controllers for autonomous aircraft, both helicopters and fixed wing, was to hire a human engineer to choose parameters for the controller," Ng says. "For example, if the helicopter is pitched forward a little more than you want, how aggressively do you want to pull back on the stick? The traditional approach was to have a person knowledgeable in aerodynamics and helicopters sit down and model that. This approach can often work, but it is a very slow design process and often doesn't perform nearly as well as modern machine learning methods."

Coates and Abbeel discovered that even the most expert human pilot's aerobic routine contains errors (or, in the language of the problem, is suboptimal). "However, repeated expert

#### **Programming**

## **Repeat Winners**

For the second year in a row, students from St. Petersburg University of Information Technology, Mechanics and Optics won the annual ACM International Collegiate Programming Contest (ICPC). With this year's victory, St. Petersburg University has now won the ACM-ICPC world championship three times in the last four years.

Known as "The Battle of the Brains," the ACM ICPC World Finals took place this year at the Royal Institute of Technology in Stockholm, Sweden. The world's top 100 university teams used open standard technology to solve 11 real-world problems involving traffic congestion, suffix-replacement grammars, and other issues, with the goal being to correctly solve the largest number of problems in the shortest amount of time.

The 33rd annual ACM ICPC, sponsored by IBM, was dominated by teams from Russia and China. This year's top 12, medal-winning teams are St. Petersburg University (Russia), which solved nine problems, followed by Tsinghua University (China), St. Petersburg State University (Russia), Saratov State University (Russia), the University of Oxford (U.K.), and Zhejiang University (China). Massachusetts Institute of Technology (U.S.) finished in seventh place, followed by Altai State Technical University (Russia), University of Warsaw (Poland), University of Waterloo (Canada), I Javakhishvili Tbilisi State University (Georgia), and Carnegie Mellon University (U.S.).

"It is clear that computational thinking, which is at the heart of the information technology revolution, is the engine that is driving innovation in these countries," says ACM President Professor Dame Wendy Hall. "As we seek to strengthen computing education and fill the talent pipeline for future workers, it is an important reminder that, while U.S. enrollment in computer science programs may have increased, we need to continue investing in programs that attract women and other underrepresented groups to this field."

demonstrations are often suboptimal in different ways," their *Learning for Control* paper noted, "suggesting that a large number of suboptimal expert demonstrations could implicitly encode the ideal trajectory the suboptimal expert is trying to demonstrate."

They discovered that merely using an arithmetic average of the states observed at any given time in the expert demonstrations would fall short of arriving at the desired trajectory, explaining that, in practice each demonstration would occur at different rates, and hence make impossible an attempt to combine states from the same time-step in each demonstration.

However, by employing the machine learning algorithm—which includes an extended Kalman filter and a dynamic programming algorithm—the researchers were able to infer the intended target trajectory and time alignment of all the demonstrations. And, while real-time variables such as the state of the air around the craft, rotor speed, actuator delays, and the behavior of the helicopter's onboard avionics contribute to an extremely complex environment that cannot be modeled accurately, these variables can be mitigated if the programming is able to make the helicopter fly the same trajectory each time. If so, the errors caused by these variables will tend to be the same, and therefore can be predicted more accurately.

In addition to the aerobatic results of the project, Coates says the ramifications for machine learning theory

go deeper. "One of the reasons people liked our paper is that it was an off-the-shelf machine learning algorithm and we solved a strange little application nobody had thought of before," Coates notes. "People know how hard this is, and to see that AI people solved this, I think has made a big impact. We had been preaching for a while that AI is the key to solving really hard problems that aren't accessible to us when we're using lots of classical methods—and if you come up with a problem and make such large strides, it really adds some weight to the argument that AI can be real and practical with algorithms that solve really hard problems."

### **Smaller, Lighter, Safer**

The future of autonomous helicopters might be even more profoundly affected by the march to increasingly powerful processors and smaller form factors.

"One way to avoid safety troubles is by making the helicopters smaller, so there are a lot of efforts going into miniaturizing the machines," says Eric Feron, professor of aerospace software engineering at Georgia Tech University, who studied autonomous helicopters while a graduate student at MIT. "That's where I think things are going now."

Coates says the breakthrough Stanford research was greatly facilitated by increased processor capability that allowed real-time instruction every 20th of a second, which was not possible even five years ago. Additionally,

the advent of microelectromechanical systems-based sensing technology, such as gyroscopes, accelerometers, and magnetometers, is leading to increased miniaturization.

Navigationally, academic researchers are now also concentrating on developing obstacle detection technology that will allow autonomous helicopters to fly safely in urban areas teeming with tall buildings, overhead wires, and light poles. Such uses are not on the near horizon, however; the ongoing safety concerns probably point to deployment in sparsely populated areas for natural resource mapping, forest firefighting, and marine search and rescue. Human-generated mapping at quarter-meter resolution can cost \$20,000 per square mile, for example, while autonomous helicopters could probably deliver the same results 10 times cheaper, says Amidi.

Georgia Tech's Feron says autonomous helicopters will continue to offer researchers an excellent platform for further research in robotics, whether the researcher is an "aeronaut" who will still be utilizing them 10 years hence, or instead testing a more universally applicable methodology on the machines, and that wider deployment will indeed follow at some point.

"The safety and reliability issues are not unworkable," Feron says. "I think it's just a matter of time." □

**Gregory Goth** is an Oakville, CT-based writer who specializes in science and technology.

© 2009 ACM 0001-0782/09/0600 \$10.00

## Awards

# American Academy Names 2009 Fellows

Computer science was well represented when the American Academy of Arts & Sciences (AAAS) recently announced the election of the 2009 class of fellows and foreign honorary members. The 212 new fellows and 19 foreign honorary members—including scholars, scientists, jurists, writers, artists, civic, corporate and philanthropic leaders—come from 28 states and 11 countries and range in age from 33 to 83. They join one of America's most prestigious honorary societies and

a center for independent policy research.

"Since 1780, the Academy has served the public good by convening leading thinkers and doers from diverse perspectives to provide practical policy solutions to the pressing issues of the day," said Leslie Berlowitz, AAAS chief executive officer. "I look forward to welcoming into the Academy these new members to help continue that tradition."

Elected in the category of computer sciences (including AI

and information technologies):

- ▶ John Seely Brown, Deloitte Center for Edge Innovation/University of Southern California
- ▶ Mary Jane Irwin, Pennsylvania State University
- ▶ Maria Klawe, Harvey Mudd College
- ▶ Ray Kurzweil, Kurzweil Technologies
- ▶ Michael Sipser, MIT

▶ Alfred Z. Spector, Google  
▶ Jennifer Widom, Stanford University.

Elected in the category of business, corporate, and philanthropic leadership:

- ▶ John Doerr, Kleiner, Perkins, Caufield & Byers.

In an email interview, John Seely Brown offered this career advice for young people: "Nurture a disposition that embraces change and that encourages you to challenge your own assumptions and having others challenge yours."

# Looking Backward and Forward

*CRA's Computing Community Consortium hosted a day-long symposium to discuss the important computing advances of the last several decades and how to sustain that track record of innovation.*

**W**HAT ARE THE major computing innovations of the recent past? How did research enable them? What advances are on the horizon, and how can they be realized? These were among the key questions addressed at an invitation-only symposium held at the Library of Congress in Washington, DC, in March.

The symposium, "Computing Research that Changed the World: Reflections and Perspectives," was organized by the Computing Research Association's (CRA's) Computing Community Consortium in cooperation with a half-dozen U.S. congressmen.

"The main goals were to explore past game-changing research in the computing fields to understand how they came about and then to take a peek at the future to see how this knowledge could be used to maximize the chances for future game-changing research," says CRA Executive Director Andrew Bernat.

"It became pretty clear that there is no foolproof way to figure out what research will turn into the big hits of tomorrow; rather, that big hits generally are a combination of independent efforts driven by curiosity and applications," Bernat says. "No one foresaw the ultimate outcomes of the initial research, so we must continue to fund a broad range of efforts in [multiple] sub-disciplines, using a variety of funding mechanisms."

The symposium's sessions included The Internet and the World Wide Web, which examined areas such as search technology and cloud computing; Evolving Foundations, which looked at the security of online information and global information networks; The Transformation of the Sciences via Computation, which covered topics such as supercomputers and the future of medicine; and Computing Everywhere!, which focused on sensing, computer graphics, and robotics.



**From left:** Daphne Koller, Stanford; Barbara Liskov, MIT; Rodney Brooks, MIT and Heartland Robotics; and Alfred Spector, Google, were among the symposium's session speakers.

Each session featured three talks and a short discussion that identified future challenges. The sessions were followed by an hour-long discussion among all the speakers, with comments from attendees, and a call to action for the future.

As for which areas of research seem particularly promising, Bernat says mobile computing will "continue to be a huge area for exploration and change as are digital media of all types. And networking will continue to boom—not just computer networking, but social networks which will help us understand the dynamics of human behavior."

Daphne Koller, a professor of computer science at Stanford University and one of the symposium's session speakers, says one of the most exciting directions in computing is the ability to use computational methods and models to analyze scientific data, particularly biomedical data.

"New biological assays are producing important data at an ever-increasing rate," Koller says. "These data have the potential of providing unprecedented

insight into basic biological processes as well as into the mechanisms and processes underlying human disease. They also have the potential of allowing us to understand the complex genetic and environmental factors that lead to differences in human phenotype, including both disease and response to drug treatment."

However, Koller adds, it's impossible to extract these insights without new computational methods. "Developing these tools is a direction where a lot of progress has been made," she says, "but much more work remains to be done."

Videos and other material from the symposium are available on the CRA Web site, and the Computing Community Consortium will host additional symposiums later this year, including one on artificial intelligence and education and another on educational data mining. C

**Bob Violino** is a writer based in Massapequa Park, NY, who covers business and technology.

© 2009 ACM 0001-0782/09/0600 \$10.00



DOI:10.1145/1516046.1516056

Eugene H. Spafford

# Privacy and Security Answering the Wrong Questions Is No Answer

*Asking the wrong questions when building and deploying systems results in systems that cannot be sufficiently protected against the threats they face.*

**F**OR OVER 50 years we have been trying to build computing systems that are trustworthy. The efforts are most notable by the lack of enduring success—and by the oftentimes spectacular security and privacy failures along the way. With each passing year (and each new threat and breach) we seem to be further away from our goals.

Consider what is present in too many organizations. Operating systems with weak controls and flaws have been widely adopted because of cost and convenience. Thus, firewalls have been deployed to put up another layer of defense against the most obvious problems. Firewalls are often configured laxly, so complex intrusion and anomaly detection tools are deployed to discover when the firewalls are penetrated. These are also imperfect, especially when insider threats are considered, so we deploy data loss detection and prevention tools. We also employ virtual machine environments intended to erect barriers against buggy implementations. These are all combined with malware detection and patch

management, yet still attacks succeed. Each time we apply a new layer, new attacks appear to defeat it.

I conjecture that one reason for these repeated failures is that we may be trying to answer the wrong questions. Asking how to make system “XYZ” secure against *all* threats is, at its core, a nonsensical question. Almost every environment and its threats are different. A system controlling a communications satellite is different from one in a bank, which in turn is different from one in an el-

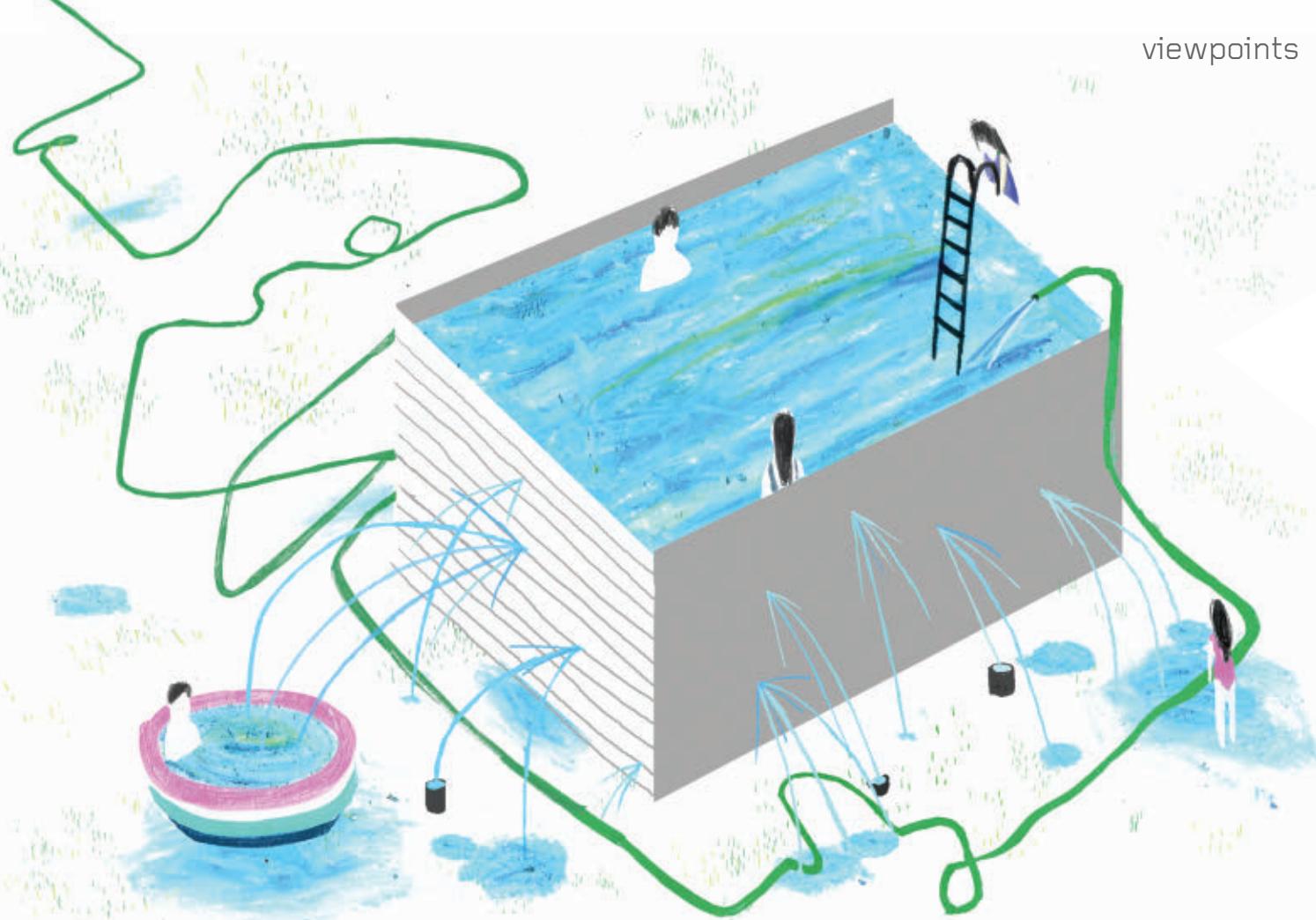
**Asking how to  
make system “XYZ”  
secure against *all*  
threats is, at its  
core, a nonsensical  
question.**

ementary school computer lab, which is different from one used to control military weapons. There are some issues in common, certainly, but the overall design and deployment should reflect the differences.

The availability and familiarity of a few common artifacts has led us to deploy them (or variants) everywhere, even to unsuitable environments. By analogy, what if everything in society was constructed of bricks because they are cheap, common, and easy to use? Imagine not only homes built of bricks, but everything else from the space shuttle to submarines to medical equipment. Thankfully, other fields have better sense and choose appropriate tools for important tasks.

A time-honored way of reinforcing a point is by means of a story told as a parable, a fairy tale, or as a joke. One classic example I tell my students:

*Two buddies leaving a tavern find a distressed and somewhat inebriated man on his hands and knees in the parking lot, apparently searching for something. They ask him what he has lost, and he replies that he has dropped his*



keys. He describes the keys, and says if the two men find them they will receive a reward. They begin to help search. Other people come by and they too are drawn into the search. Soon, there is a crowd combing the lot, with an air of competition to see who will be the first to find the keys. Periodically someone informs the crowd of the discovery of a coin or a particularly interesting piece of rock.

After a while, one in the crowd stands up and inquires of the fellow who lost his keys, "Say, are you sure you lost your keys out here in the lot?" To which the man replies, "No. I lost them in the alley." Everyone stops to stare at the man. "Well, why the heck are you searching for them here in the parking lot?" someone exclaimed. To which the man replied, "Well, the light is so much better here. And besides, now I have such good company!"

There are many lessons that can be inferred from this story, but the one I stress with my students is that if they don't properly define the problem, ask the right questions, and search in the proper places, they may have good company and funding, but they

shouldn't expect to find what they are really seeking.<sup>a</sup>

So it is in research—especially in cyber security and privacy. We have people seeking answers to the wrong questions, often because that is where "the light is better" and there seems to be a bigger crowd around them. Until we start asking questions that better address the problems that really need to be solved, we shouldn't expect to see progress. Here are a few examples of misleading questions:

- ▶ How do I secure my commodity operating system against all threats?
- ▶ How do I protect my system with an intrusion-detection system, data loss and prevention tools, firewalls, and other techniques?
- ▶ How do I find coding flaws in the system I am using so I can patch them?
- ▶ How do we build multilevel secure systems?

Each of these questions implies it can be answered in a positive, mean-

ingful manner. That is not necessarily the case.

We have generally failed to understand that when we build and deploy systems they are used in a variety of environments, facing different threats. There is no perfect security in any real system—hardware fails, people make mistakes, and attacks outside our expectations may defeat our protection mechanisms. If an attacker is sufficiently motivated and has enough resources (including time), every system can be defeated in some manner.<sup>b</sup> If the attacker doesn't care if the defeat is noticed, it may reduce the work factor involved; as an obvious example, an assured denial-of-service attack can be accomplished with enough nuclear weapons. The goal in the *practice* of security is to construct sufficient defenses against the likely threats in such a

<sup>a</sup> Another story that resonates with my students is <http://spaf.cerias.purdue.edu/Archive/racehorse.html>.

<sup>b</sup> There are many books on this topic, and the basic premise is at the heart of nearly every big heist movie, including *Ocean's 11*, *The Italian Job*, and *The Thomas Crown Affair*. For some interesting, real-life examples outside computing, I recommend the book *Spycraft* by Robert Wallace and H. Keith Melton.



# ACM Journal on Computing and Cultural Heritage



◆◆◆◆◆

**JOCCH** publishes papers of significant and lasting value in all areas relating to the use of ICT in support of Cultural Heritage, seeking to combine the best of computing science with real attention to any aspect of the cultural heritage sector.

◆◆◆◆◆

[www.acm.org/jocch](http://www.acm.org/jocch)  
[www.acm.org/subscribe](http://www.acm.org/subscribe)



Association for  
Computing Machinery

way as to reduce the risk of compromise to an acceptable level; if the attack can be made to cost far more than the perceived gain resulting from its success, then that is usually sufficient.

By asking the wrong questions—such as how to patch or modify existing items rather than ask what is appropriate to build or acquire—we end up with systems that cannot be adequately protected against the threats they face. Few current systems are designed according to known security practices,<sup>c</sup> nor are they operated within an appropriate policy regime. Without understanding the risks involved, management seeks to “add on” security technology to the current infrastructure, which may add new vulnerabilities.

The costs of replacing existing systems with different ones requiring new training seems so daunting that it is seldom considered, even by organizations that face prospects of catastrophic loss. There is so much legacy code that developers and customers alike believe they cannot afford to move to something else. Thus, the market tends toward “add on” solutions and patches rather than fundamental reengineering. Significant research funding is applied to tinkering with current platforms rather than addressing the more fundamental issues. Instead of asking “How do we design and build systems that are secure in a given threat environment?” and “What tools and programming constructs should we be using to produce systems that do not exhibit easily exploited flaws?” we, as a community, continue to ask the wrong questions.

Note that I am not arguing against standards, per se. Standards are important for interoperability and innovation. However, standards are best applied at the interfaces so as to allow innovation and good engineering practice to take place *inside*. I am also not overlooking the potential expense. Creating new systems, training developers, and developing new code bases might be costly, but

<sup>c</sup> There are many fine works on security engineering, including Ross Anderson's opus of that title. If we return to the fundamentals, tried-and-true design principles were articulated by Jerome H. Saltzer and Michael D. Schroeder in “The Protection of Information in Computer Systems,” republished in *Communications of the ACM* 17, 7 (July 1974) but few systems are designed using these principles.

only initially—given current losses and trends, this approach would eventually reduce costs in many environments.

Robert H. (Bob) Courtney Jr., one of the first computer security professionals and an early recipient of the NIST/NCSC National Computer Systems Security Award articulated three “laws” for those who seek to build secure, operational computational artifacts:<sup>d</sup>

- Nothing useful can be said about the security of a mechanism except in the context of a specific application and environment.

- Never spend more mitigating a risk than tolerating what it will cost you.

- There are management solutions to technical problems but no technical solutions to management problems.

Although not everyone will agree with these three laws, they provide a good starting point for thinking about the practice of information security. The questions we should be asking are not about how to secure system “XYZ,” but whether “XYZ” is appropriate for use in the environment at hand. Can it be configured and protected against the expected threats to a level that matches our risk tolerance? What policies and procedures need to be put in place to augment the technology? What is the true value of what we are protecting? Do we even know what we are protecting?<sup>e</sup>

As researchers and practitioners, we need to stop looking for solutions where the light is good and people seem to be gathered. Consider a quote I have been using recently: “Insanity is doing the same thing over and over again while expecting different results.”<sup>f</sup> Asking the wrong questions repeatedly is not only hindering us from making real progress but may even be considered insane.

So, what questions are you trying to answer?

<sup>d</sup> My thanks to William Hugh Murray for his restatement of Courtney's Laws.

<sup>e</sup> Many firms do not understand the value of what they are protecting or where it is located; see <http://snipurl.com/sec-econ>.

<sup>f</sup> This quote is widely attributed to Albert Einstein and to John Dryden. I have been unable to find a definitive source for it, however.

**Eugene H. Spafford** (spaf@cerias.purdue.edu) is a professor of computer science and the executive director of the Center for Education and Research in Information Assurance and Security (CERIAS) at Purdue University.

# Inside Risks

## Reducing Risks of Implantable Medical Devices

*A prescription to improve security and privacy of pervasive health care.*

**M**ILLIONS OF PATIENTS benefit from programmable, implantable medical devices (IMDs) that treat chronic ailments such as cardiac arrhythmia,<sup>6</sup> diabetes, and Parkinson's disease with various combinations of electrical therapy and drug infusion. Modern IMDs rely on radio communication for diagnostic and therapeutic functions—allowing health-care providers to remotely monitor patients' vital signs via the Web and to give continuous rather than periodic care. However, the convergence of medicine with radio communication and Internet connectivity exposes these devices not only to safety and effectiveness risks, but also to security and privacy risks. IMD security risks have greater direct consequences than security risks of desktop computing. Moreover, IMDs contain sensitive information with privacy risks more difficult to mitigate than that of electronic health records or pharmacy databases. This column explains the impact of these risks on patient care, and makes recommendations for legislation, regulation, and technology to improve security and privacy of IMDs.

### Consequences and Causes: Security Risks

The consequences of an insecure IMD can be fatal. However, it is fair to ask whether intentional IMD malfunctions represent a genuine threat. Unfortunately, there are people who



From left, Benjamin Ransford (University of Massachusetts), Daniel Halperin (University of Washington), Benessa Defend (University of Massachusetts), and Shane Clark (University of Massachusetts) worked to uncover security flaws in implantable medical devices.

cause patients harm. In 1982, someone deliberately laced Tylenol capsules with cyanide and placed the contaminated products on store shelves in the Chicago area. This unsolved crime led to seven confirmed deaths, a recall of an estimated 31 million bottles of Tylenol, and a rethinking of security for packaging medicine in a tamper-evident manner. Today, IMDs appear to offer a similar opportunity to other depraved people. While there are no reported incidents of deliberate interference, this can change at any time. The global reach of the Internet and the prevalence and inter-

mingling of radio communications expose IMDs to historically open environments with difficult to control perimeters.<sup>3,4</sup> For instance, vandals caused seizures in photosensitive individuals by posting flashing animations on a Web-based epilepsy support group.<sup>1</sup>

Knowing that such vandals will always exist, the next question is whether genuine security risks exist. What could possibly go wrong by allowing an IMD to communicate over great distances with radio and then mixing in Internet-based services? It does not require much sophistication

to think of numerous ways to cause intentional malfunctions in an IMD. Few desktop computers have failures as consequential as that of an IMD. Intentional malfunctions can actually kill people, and are more difficult to prevent than accidental malfunctions. For instance, lifesaving therapies were silently modified and disabled via radio communication on an implantable defibrillator that had passed premarket approval by regulators.<sup>3</sup> In my research lab, the same device was reprogrammed with an unauthenticated radio-based command to induce a shock that causes ventricular fibrillation (a fatal heart rhythm).

Manufacturers point out that IMDs have used radio communication for decades, and that they are not aware of any unreported security problems. Spam and viruses were also not prevalent on the Internet during its many-decade nascent period. Firewalls, encryption, and proprietary techniques did not stop the eventual onslaught. It would be foolish to assume IMDs are any more immune to malware. For instance, if malware were to cause an IMD to continuously wake from power-saving mode, the battery would wear out quickly. The malware creator need not be physically present, but could expose a patient to risks of unnecessary surgery that could lead to infection or death. Much like Macintosh users can take comfort in that most current malware takes aim at the Windows

platform, patients can take comfort in that IMDs seldom rely on such widely targeted software for now.

### **Consequences and Causes: Privacy**

A second risk is violation of patient privacy. Today's IMDs contain detailed medical information and sensory data (including vital signs, patient name, date of birth, therapies, and medical diagnosis). Data can be read from an IMD by passively listening to radio communication. With newer IMDs providing nominal read ranges of several meters, eavesdropping will become easier. The privacy risks are similar to that of online medical records.

### **Remedies**

Improving IMD security and privacy requires a proper mix of technology and regulation.

### **Remedy: Technology**

Technological approaches to improving IMD security and privacy include judicious use of cryptography and limiting unnecessary exposure to would-be hackers. IMDs that rely on radio communication or have pathways to the Internet must resist a determined adversary.<sup>5</sup> IMDs can last upward of 20 years, and doctors are unlikely to surgically replace an IMD just because a less-vulnerable one becomes available. Thus, technologists must think 20 to 25 years out. Cryptographic systems

available today may not last 25 years.

It is tempting to consider software updates as a remedy for maintaining the security of IMDs. Because software updates can lead to unexpected malfunctions with serious consequences, pacemaker and defibrillator patients make an appointment with a health-care provider to receive firmware updates in a clinic. Thus, it could take too long to patch a security hole.

Beyond cryptography, several steps could reduce exposure to potential misuse. When and where should an IMD permit radio-based, remote reprogramming of therapies (such as changing the magnitude of defibrillation shocks)? When and where should an IMD permit radio-based, remote collection of telemetry (for example, vital signs)? Well-designed cryptographic authentication and authorization make these two questions solvable. Does a pacemaker really need to accept requests for reprogramming and telemetry in all locations from street corners to subway stations? The answer is no. Limit unnecessary exposure.

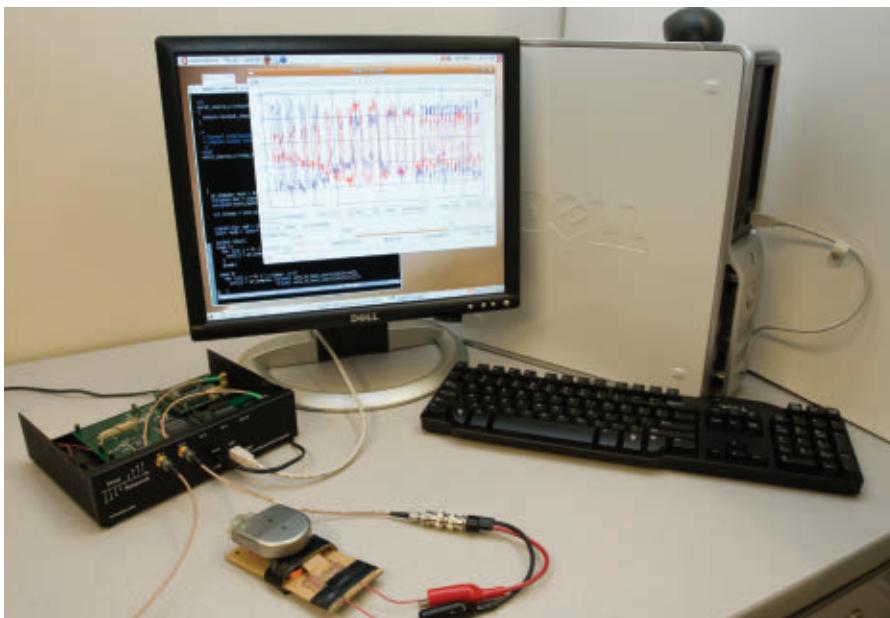
### **Remedy: Regulation**

Premarket approval for life-sustaining IMDs should explicitly evaluate security and privacy—leveraging the body of knowledge from secure systems and security metrics communities. Manufacturers have already deployed hundreds of thousands of IMDs without voluntarily including reasonable technology to prevent the unauthorized induction of a fatal heart rhythm. Thus, future regulation should provide incentives for improved security and privacy in IMDs.

Regulatory aspects of protecting privacy are more complicated, especially in the United States. Although the U.S. Food and Drug Administration has acknowledged deleterious effects of privacy violations on patient health,<sup>2</sup> there is no ongoing process or explicit requirement that a manufacturer demonstrate adequate privacy protection. The FDA has no legal remit from Congress to directly regulate privacy (the FDA does not administer HIPAA privacy regulations).

### **Call to Action**

My call to action consists of two parts



**Equipment used to attack an implantable cardiac defibrillator (ICD).**

## Improving IMD security and privacy requires a proper mix of technology and regulation.

legislation, one part regulation, and one part technology.

First, legislators should mandate stronger security during premarket approval of life-sustaining IMDs that rely on either radio communication or computer networking. Action at premarket approval is crucial because unnecessary surgical replacement directly exposes patients to risk of infection and death. Moreover, the threat models and risk retention chosen by the manufacturer should be made public so that health-care providers and patients can make informed decisions when selecting an IMD. Legislation should avoid mandating specific technical approaches, but instead should provide incentives and penalties for manufacturers to improve IMD security.

Second, legislators should give regulators the authority to require adequate privacy controls before allowing an IMD to reach the market. The FDA writes that privacy violations can affect patient health,<sup>2</sup> and yet the FDA has no direct authority to regulate privacy of medical devices. IMDs increasingly store large amounts of sensitive medical information and fixing a privacy flaw after deployment is especially difficult on an IMD. Moreover, security and privacy are often intertwined. Inadequate security can lead to inadequate privacy, and inadequate privacy can lead to inadequate security. Thus, device regulators have the unique vantage point for not only determining safety and effectiveness, but also determining security and privacy.

Third, regulators such as the FDA should draw upon industry, the health-care community, and academics to conduct a thorough and open review of security and privacy metrics

for IMDs. Today's guidelines are so ambiguous that an implantable cardioverter defibrillator with no apparent authentication whatsoever has been implanted in hundreds of thousands of patients.<sup>3</sup>

Fourth, technologists should ensure that IMDs do not continue to repeat the mistakes of history by underestimating the adversary, using outdated threat models, and neglecting to use cryptographic controls.<sup>5</sup> In addition, technologists should not dismiss the importance of usable security and human factors.

### Conclusion

There is no doubt that IMDs save lives. Patients prescribed such devices are much safer with the device than without, but IMDs are no more immune to security and privacy risks than any other computing device. Yet the consequences for IMD patients can be fatal. Tragically, it took seven cyanide poisonings in the 1982 Chicago Tylenol poisoning case for the pharmaceutical industry to redesign the physical security of its product distribution to resist tampering by a determined adversary. The security and privacy problems of IMDs are obvious, and the consequences just as deadly. We'd better get it right today, because surgically replacing an insecure IMD is much more difficult than an automated Windows update. □

### References

1. Epilepsy Foundation. Epilepsy Foundation Takes Action Against Hackers. March 31, 2008; [http://www.epilepsyfoundation.org/aboutus/pressroom/action\\_against\\_hackers.cfm](http://www.epilepsyfoundation.org/aboutus/pressroom/action_against_hackers.cfm).
2. FDA Evaluation of Automatic Class III Designation VeriChip™ Health Information Microtransponder System, October 2004; <http://www.sec.gov/Archives/edgar/data/924642/000106880004000587/ex99p2.txt>.
3. Halperin, D. et al. Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In *Proceedings of the 29th Annual IEEE Symposium on Security and Privacy*, May 2008.
4. Halperin, D. et al. Security and privacy for implantable medical devices. In *IEEE Pervasive Computing, Special Issue on Implantable Electronics* (Jan. 2008).
5. Schneier, B. Security in the real world: How to evaluate security technology. *Computer Security Journal* 15, 4 (Apr. 1999); <http://www.schneier.com/essay-031.html>.
6. Webster, J.G., Ed. *Design of Cardiac Pacemakers*. IEEE Press, 1995.

**Kevin Fu** ([kevinfu@cs.umass.edu](mailto:kevinfu@cs.umass.edu)) is an assistant professor of computer science at the University of Massachusetts Amherst.

This work was supported by NSF grant CNS-0831244.

Copyright held by author.

## Calendar of Events

### June 16–18

Conference on the Future of the Internet 2009, Seoul Republic of Korea, Contact: Craig Partridge, Phone: 517-324-3425, Email: [craig@bbn.com](mailto:craig@bbn.com)

### June 19–20

International Symposium on Memory Management, Dublin, Ireland, Sponsored: SIGPLAN, Contact: Elliot K Kolodner, Email: [kolodner@il.ibm.com](mailto:kolodner@il.ibm.com)

### June 19–20

ACM SIGPLAN/SIGBED 2009 Conference on Languages, Compilers, and Tools for Embedded Systems, Dublin, Ireland, Sponsored: SIGPLAN, Contact: Christoph Kirsch, Email: [ck@cs.uni-salzburg.at](mailto:ck@cs.uni-salzburg.at)

### June 20–24

The 36th Annual International Symposium on Computer Architecture, Austin, TX, Sponsored: SIGARCH, Contact: Stephen W. Keckler, Phone: 512-471-9763, Email: [sheckler@cs.utexas.edu](mailto:sheckler@cs.utexas.edu)

### June 22

Fourth International Workshop on Mobility in the Evolving Internet Architecture, Krakow, Poland, Contact: Prof. Xiaoming, Email: [fu@cs.uni-goettingen.de](mailto:fu@cs.uni-goettingen.de)

### June 22–23

Second International Workshop on Future Multimedia Networking, Coimbra, Portugal, Contact: Eduardo Cerquiera, Email: [ecelho@dei.uc.pt](mailto:ecelho@dei.uc.pt)

### June 22–25

23rd ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation, Lake Placid, NY, Contact: Carl Tropper, Email: [carl@cs.mcgill.ca](mailto:carl@cs.mcgill.ca)

### June 23–26

12th International Symposium on Component Based Software Engineering, East Stroudsburg, PA, Sponsored: SIGSOFT, Contact: Christine Hofmeister, Email: [Christine.hofmeister@gmail.com](mailto:Christine.hofmeister@gmail.com)



DOI:10.1145/1516046.1516054

Peter J. Denning

## The Profession of IT Beyond Computational Thinking

*If we are not careful, our fascination with “computational thinking” may lead us back into the trap we are trying to escape.*

**I**N THE MIDST of our struggle to better articulate why computing is so much broader than programming, a movement of sorts has emerged. It is being called “computational thinking.”<sup>8</sup> The U.S. National Science Foundation’s Computer and Information Science and Engineering (CISE) directorate has asked most proposers, especially those in its CPATH initiative, to include a discussion of how their projects advance computational thinking. Carnegie Mellon University’s Center for Computational Thinking says, “It is nearly impossible to do research in any scientific or engineering discipline without an ability to think computationally....[We] advocate for the widespread use of computational thinking to improve people’s lives.”<sup>1</sup>

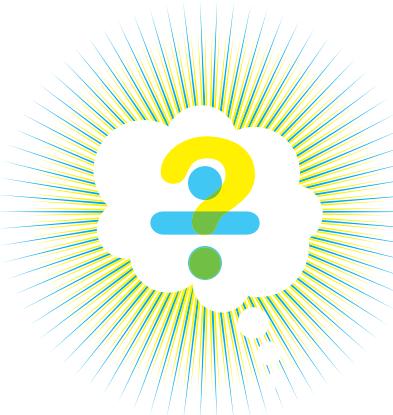
Computational thinking is seen by its adherents as a novel way to say what the core of the field is about, a lever to reverse the decline of enrollments, and a rationale for accepting computer science as a legitimate field of science. This movement is driven by four main concerns:

- ▶ Bringing computer science to the table of science (as partner, not programmer).
- ▶ Finding ways to make computer science a more attractive field for students to major in and for other sciences to collaborate with.
- ▶ Resurrecting ongoing inquiry into

the deep questions of the field.<sup>6,9</sup>

- ▶ Showing that computation is fundamental, and often unavoidable, in most endeavors—a desire to proselytize.

Since starting a stint at NASA-Ames in 1983, I have been heavily involved with computational science and I have devoted a substantial part of my own career to advancing these objectives. Since



2003 I have advocated a great-principles approach to the perennially open question, “What is computer science?”<sup>4</sup>

Yet I am uneasy. I am concerned that the computational thinking movement reinforces a narrow view of the field and will not sell well with the other sciences or with the people we want to attract. I worry that we are not getting out of the box, but are merely repackaging it with new paper and a fresh ribbon.

In this column, I will examine two key questions:

- ▶ Is computational thinking a unique and distinctive characterization of computer science?

- ▶ Is computational thinking an adequate characterization of computer science?

My own conclusion is that both answers are no. I will suggest that a principles-based framework answers both questions yes. We are custodians of a deep and powerful discourse: Let’s not hide it with an inadequate name.

### What is Computational Thinking?

Computational thinking has a long history within computer science. Known in the 1950s and 1960s as “algorithmic thinking,” it means a mental orientation to formulating problems as conversions of some input to an output and looking for algorithms to perform the conversions.

Today the term has been expanded to include thinking with many levels of abstractions, use of mathematics to develop algorithms, and examining how well a solution scales across different sizes of problems.<sup>1</sup>

### Is Computational Thinking Unique to Computer Science?

In the 1940s, John von Neumann wrote prolifically on how computers would be not just a tool for helping science, but a way of doing science.

As early as 1975, Physics Nobel Laureate Ken Wilson promoted the idea that simulation and computation

were a way to do science that was not previously available. Wilson's Nobel Prize was based on breakthroughs he achieved in creating computational models whose simulations produced radical new understandings of phase changes in materials. In the early 1980s, Wilson joined with other leading scientists in many fields to advocate that the grand challenges of science could be cracked by computation and that the government could accelerate the process by supporting a network of supercomputing centers.<sup>7</sup> They argued that computation had become a third leg of science, joining the traditional legs of theory and experiment. The term "computational thinking" was common in their discussions.

The computational sciences movement eventually grew into a huge interagency initiative in high-performance computing, and culminated in the U.S. Congress passing a law funding a high-performance computing initiative in 1991.

This movement validated the notion that computation (and computational thinking) is essential to the advancement of science. It generated a powerful political movement that codified this notion into a U.S. federal law.

It is important to notice that this movement originated with the leaders of the physical and life sciences. Com-

## Computation is unavoidable not only in the method of study, but in what is studied.

puter science was present but was not a key player. Computer scientists, in fact, resisted participation until NSF CISE and DARPA set up research programs open only to those collaborating with other sciences.

In the middle 1980s, Ken Wilson advocated the formation of departments of computational science in universities. He carefully distinguished them from computer science. The term "computational science" was chosen to avoid confusion with computer science.

Thus, computational science is seen in the other sciences not as a notion that flows out of computer science, but as a notion that flows from science itself. Computational thinking is seen as a characteristic of this way of science. It is not seen as a distinctive feature of computer science.

Therefore, it is unwise to pin our hopes on computational thinking as a way of telling people about the unique character of computer science. We need some other way to do that.

The sentiment that computational thinking is a recent insight into the true nature of computer science ignores the venerable history of computational thinking in computer science and in all the sciences. Computer science is a science in its own right (see the sidebar "Computer Science as Science").

### Is Computational Thinking Adequate for Computer Science?

In 1936 Alan Turing defined what it means to compute a number. He offered a model of a computing machine and showed that the machines were universal (one could simulate another). He then used his theory to settle a century-old "decision problem" of mathematics, whether there is a by-inspection method to tell if a set of decision rules can terminate with a decision in a finite number of moves. He showed that the "decision problem" was not computable and argued that the very act of inspecting is inherently computational: not even inspectors can avoid computation. Computation is universal and unavoidable. His paper truly was the birth of computer science.

The modern formulations of science

# Computer Science as Science

Since its beginnings in the late 1930s, computer science has been a unique combination of math, engineering, and science. It is not one, but all three. Major subsets form legitimate fields of math, engineering, or science. But if you focus on a single subset, you cannot express the uniqueness of the field.

The term "computer science" traces back to the writings of John von Neumann, who believed that the architecture of machines and applications could be put on a rigorous scientific basis.

Until about 1990, the emphasis within the field was developing and advancing the technology. Building reliable computers within a

networking infrastructure was a grand challenge that took many years. Now that this has been accomplished, we are increasingly able to emphasize the experimental method and reinvigorate our image as a science. Our many partnerships with other sciences including biology, physics, astronomy, materials science, economics, cognitive science, and sociology, have led to amazing innovations.

These collaborations have uncovered questions in the other fields about whether computer science is legitimately science. Many see computer people as engineers implementing principles they did not discover rather than

equal partners in the search for new principles. So it matters whether computer science qualifies as a full-fledged science. Whether a field is seen as a science depends on its satisfying six criteria:<sup>5</sup>

- ▶ Has an organized body of knowledge
- ▶ Results are reproducible
- ▶ Has well developed experimental methods
- ▶ Enables predictions, including surprises
- ▶ Offers hypotheses open to falsification
- ▶ Deals with natural objects

Computer science easily passes the first five of these tests, so the debate has tended to center on the last. During the past decade, prominent

scientists in other fields have discovered natural information processes—affirming the sixth criterion.<sup>3</sup> The older definition of computer science as "the study of phenomena surrounding computers," which dates back to Alan Perlis, George Forsythe, and Allen Newell around 1970, is giving way to "the study of information processes, natural and artificial." The shift from computer as object of study to computer as tool is enabling us to revisit the deep questions of our field in the new light of computation as a lens through which to see the world. The most fundamental of these questions is: What is computation?<sup>6,9</sup>

recognize the same truth when they say that computation is an essential method of doing science. In fact, a growing number of scientists are now saying that information processes occur naturally (for example, DNA transcription) and that computation is needed to understand and eventually control them.<sup>3</sup> So computation is unavoidable not only in the method of study, but in what is studied.

This is a subtle but important distinction. Computation is present in nature even when scientists are not observing it or thinking about it. Computation is more fundamental than computational thinking. For this reason alone, computational thinking seems like an inadequate characterization of computer science.

A number of us developed a great principles framework that exposes the fundamental scientific principles of computing<sup>4,6</sup> (see the sidebar “The Great Principles Framework”). This framework interprets computer science as the study of fundamental properties of information processes, both natural and artificial. Computers are the tool, not the object of study. Computation pervades everyday life.<sup>2</sup>

The great principles framework reveals that there is something even more fundamental than an algorithm: the representation. Representations convey information. A computation is an evolving representation and an algorithm is a representation of a method to control the evolution.

In this framework, computational thinking is not a principle; it is a practice. A practice is a way of doing things

## The real value of computer science is in the offers we are able to make from our expertise, which is founded in a rich and deep discourse.

at which we can develop various levels of skill. Computational thinking is one of several key practices at which every computer scientist should be competent (see the sidebar “The Great Principles Framework”). It shortchanges computer science to try to characterize the field by mentioning only one essential practice without mentioning the others or the principles of the field.

### Conclusion

Computation is widely accepted as a lens for looking at the world. We do not need to sell that idea. Computational thinking is one of the key practices of computer science. But it is not unique to computing and is not adequate to portray the whole of the field.

In the 1960s and 1970s we allowed, and even encouraged, the perception “CS = programming,” which is now to our dismay widely accepted outside the field and is connected with our inabil-

ity to take care of the concerns listed at the beginning of this column. But given the outside perception, computational thinking is all too easily seen as a repackaging—a change of appearance but not of substance. Do we really want to replace that older notion with “CS = computational thinking”? A colleague from another field recently said to me: “You computer scientists are hungry! First you wanted us to take your courses on literacy and fluency. Now you want us to think like you!”

I suggest that the real value of computer science is in the offers we are able to make from our expertise, which is founded in a rich and deep discourse. We are valued at the table when we help the others solve problems they care about. We are most valued not for our computational thinking, but for our computational doing. C

### References

1. Carnegie Mellon University Center for Computational Thinking: <http://www.cs.cmu.edu/~CompThink>.
2. Computer Science Unplugged Web site: <http://csunplugged.org>.
3. Denning, P. Computing is a natural science. *Commun. ACM* 50, 7 (July 2007), 13–18.
4. Denning, P. Great principles of computing. *Commun. ACM* 46, 11 (Nov. 2003), 15–20.
5. Denning, P. Is computer science science? *Commun. ACM* 48, 4 (Apr. 2005), 27–31.
6. Great Principles of Computing Web site: <http://greatprinciples.org>.
7. Wilson, K.G. Grand challenges to computational science. In *Future Generation Computer Systems*. Elsevier, 1989, 171–189.
8. Wing, J. Computational thinking. *Commun. ACM* 49, 3 (Mar. 2006), 33–35.
9. Wing, J. Five deep questions in computing. *Commun. ACM* 51, 1 (Jan. 2008), 58–60.

**Peter J. Denning** ([pjd@nps.edu](mailto:pjd@nps.edu)) is the director of the Cebrowski Institute for Information Innovation and Superiority at the Naval Postgraduate School in Monterey, CA, and is a past president of ACM.

Copyright held by author.

# The Great Principles Framework

The Great Principles (GP) framework is a way to express computer science as a field of science based on deep and enduring fundamental principles.<sup>3,4,6</sup> The framework has two parts: core principles and core practices.

The core principles are statements and stories about the immutable laws and recurrences that shape and constrain all computing

technologies. They can be grouped into seven categories:

- Computation
- Communication
- Coordination
- Recollection
- Automation
- Evaluation
- Design

These are not mutually exclusive groups of principles, but windows that bring particular perspectives about

computing. The Internet, for example, is a technology that draws its operating principles primarily from communication, coordination, and recollection, and its architecture from design and evaluation.

The core practices are areas of skill and ability at which computing people can display various levels of performance such as beginner, competent, and expert. There are four core

practices:

- Programming
- Engineering of systems
- Modeling
- Applying

Computational thinking can be seen either as a style of thought that runs through the practices or as a fifth practice. It is the ability to interpret the world as algorithmically controlled conversions of inputs to outputs.

## Viewpoint

# Why “Open Source” Misses the Point of Free Software

*Decoding the important differences in terminology, underlying philosophy, and value systems between two similar categories of software.*

**W**HEN WE CALL software “free,” we mean it respects the users’ essential freedoms: the freedom to run it, to study and change it, and to redistribute copies with or without changes (see <http://www.gnu.org/philosophy/free-sw.html>). This is a matter of freedom, not price, so think of “free speech,” not “free beer.”

These freedoms are vitally important. They are essential, not just for the individual users’ sake, but because they promote social solidarity—that is, sharing and cooperation. They become even more important as more aspects of our culture and life activities are digitized. In a world of digital sounds, images, and words, free software increasingly equates with freedom in general.

Tens of millions of people around the world now use free software; the schools in regions of India and Spain now teach all students to use the free GNU/Linux operating system (see <http://www.gnu.org/gnu/linux-and-gnu.html>). But most of these users have never heard of the ethical reasons for which we developed this system and built the free software community, because today this system and community are more often described as “open source,” and attributed to a different philosophy in which these freedoms are hardly mentioned.

The free software movement has campaigned for computer users’ freedom since 1983. In 1984 we launched



the development of the free operating system GNU, so we could avoid the non-free operating systems that deny freedom to their users. During the 1980s, we developed most of the essential components of such a system, as well as the GNU General Public License (see <http://www.gnu.org/licenses/gpl.html>), a license designed specifically to protect freedom for all users of a program.

However, not all of the users and developers of free software agreed with the goals of the free software movement. In 1998, a part of the free software community splintered off and began campaigning in the name of “open source.” The term was originally proposed to avoid a possible misunderstanding of the term “free software,” but it soon became associated with philosophical views quite different from those of the free software movement.

Some of the proponents of “open source” considered it a marketing campaign for free software, which would

appeal to business executives by citing practical benefits, while avoiding ideas of right and wrong they might not like to hear. Other proponents flatly rejected the free software movement’s ethical and social values. Whichever their views, when campaigning for “open source” they did not cite or advocate those values. The term “open source” quickly became associated with the practice of citing only practical values, such as making powerful, reliable software. Most of the supporters of “open source” have come to it since then, and that practice is what they take it to mean.

Nearly all open source software is free software; the two terms describe almost the same category of software. But they stand for views based on fundamentally different values. Open source is a development methodology; free software is a social movement. For the free software movement, free software is an ethical imperative, because only free software respects the users’ freedom. By contrast, the philosophy of open source considers issues in terms of how to make software “better”—in a practical sense only. It says that non-free software is a suboptimal solution. For the free software movement, however, non-free software is a social problem, and moving to free software is the solution.

Free software. Open source. If it’s the same software, does it matter which name you use? Yes, because different words convey different ideas.

While a free program by any other name would give you the same freedom today, establishing freedom in a lasting way depends above all on teaching people to value freedom. If you want to help do this, it is essential to speak about “free software.”

We in the free software movement don’t think of the open source camp as an enemy; the enemy is proprietary (non-free) software. But we want people to know we stand for freedom, so we do not accept being misidentified as open source supporters.

### **Common Misunderstandings of “Free Software” and “Open Source”**

The term “free software” has a problem of misinterpretation: an unintended meaning, “software you can get for zero price,” fits the term just as well as the intended meaning, “software that gives the user certain freedoms.” We address this problem by publishing the definition of free software, and by saying “Think of free speech, not free beer.” This is not a perfect solution; it cannot completely eliminate the problem. An unambiguous, correct term would be better, if it didn’t have other problems.

Unfortunately, all the alternatives in English have problems of their own. We’ve looked at many alternatives that people have suggested, but none is so clearly correct that switching to it would be a good idea. Every proposed replacement for “free software” has some kind of semantic problem—and this includes “open source software.”

The official definition of “open source software,” which is published by the Open Source Initiative (see <http://opensource.org/docs/osd>) and too long to cite here, was derived indirectly from our criteria for free software. It is not

the same; it is a little looser in some respects, so open source supporters have accepted a few licenses that we consider unacceptably restrictive of the users. Nonetheless, it is fairly close to our definition in practice.

However, the obvious meaning for the expression “open source software” is “You can look at the source code,” and most people seem to think that’s what it means. That is a much weaker criterion than free software, and much weaker than the official definition of open source. It includes many programs that are neither free nor open source. Since that obvious meaning for “open source” is not the meaning that its advocates intend, the result is that most people misunderstand the term. Here is how writer Neal Stephenson defined “open source”: *Linux is “open source” software meaning, simply, that anyone can get copies of its source code files.*

I don’t think Stephenson deliberately sought to reject or dispute the “official” definition. I think he simply applied the conventions of the English language to come up with a meaning for the term. The state of Kansas published a similar definition: *Make use of open-source software (OSS). OSS is software for which the source code is freely and publicly available, though the specific licensing agreements vary as to what one is allowed to do with that code.*

Open source supporters try to deal with this by pointing to their official definition, but that corrective approach is less effective for them than it is for us. The term “free software” has two natural meanings, one of which is the intended meaning, so a person who has grasped the idea of “free speech, not free beer” will not get it wrong again. But “open source” has only one natural meaning, which is different from the meaning its supporters intend. So there is no succinct way to explain and justify the official definition of “open source.” That makes for worse confusion.

Another common misunderstanding of “open source” is the idea that it means “not using the GNU GPL.” It tends to accompany a misunderstanding of “free software,” equating it to “GPL-covered software.” These are equally mistaken, since the GNU GPL is considered an open source license, and most of the open source licenses are

considered free software licenses.

### **Different Values Can Lead to Similar Conclusions... But Not Always**

Radical groups in the 1960s had a reputation for factionalism: some organizations split because of disagreements on details of strategy, and the two resultant groups treated each other as enemies despite having similar basic goals and values. The right wing made much of this, and used it to criticize the entire left.

Some try to disparage the free software movement by comparing our disagreement with open source to the disagreements of those radical groups. They have it backward. We disagree with the open source camp on the basic goals and values, but their views and ours lead in many cases to the same practical behavior—such as developing free software.

As a result, people from the free software movement and the open source camp often work together on practical projects such as software development. It is remarkable that such different philosophical views can so often motivate different people to participate in the same projects. Nonetheless, these views are very different, and there are situations where they lead to very different actions.

The idea of open source is that allowing users to change and redistribute the software will make it more powerful and reliable. But this is not guaranteed. Developers of proprietary software are not necessarily incompetent. Sometimes they produce a program that is powerful and reliable, even though it does not respect the users’ freedom. How will free software activists and open source enthusiasts react to that?

A pure open source enthusiast, one that is not at all influenced by the ideals of free software, will say, “I am surprised you were able to make the program work so well without using our development model, but you did. How can I get a copy?” This attitude will reward schemes that take away our freedom, leading to its loss.

The free software activist will say, “Your program is very attractive, but not at the price of my freedom. So I have to do without it. Instead I will support a project to develop a free replacement.”

**Open source is a development methodology; free software is a social movement.**

If we value our freedom, we can act to maintain and defend it.

### **Powerful, Reliable Software Can Be Bad**

The idea that we want software to be powerful and reliable comes from the supposition that the software is designed to serve its users. If it is powerful and reliable, that means it serves them better.

But software can only be said to serve its users if it respects their freedom. What if the software is designed to put chains on its users? Then powerlessness only means the chains are more constricting, and reliability that they are harder to remove. Malicious features, such as spying on the users, restricting the users, back doors, and imposed upgrades are common in proprietary software, and some open source supporters want to do likewise.

Under the pressure of the movie and record companies, software for individuals to use is increasingly designed specifically to restrict them. This malicious feature is known as DRM, or Digital Restrictions Management (see [DefectiveByDesign.org](http://DefectiveByDesign.org)), and it is the antithesis in spirit of the freedom that free software aims to provide. And not just in spirit: since the goal of DRM is to trample your freedom, DRM developers try to make it difficult, impossible, or even illegal for you to change the software that implements the DRM.

Yet some open source supporters have proposed “open source DRM” software. Their idea is that by publishing the source code of programs designed to restrict your access to encrypted media, and allowing others to change it, they will produce more powerful and reliable software for restricting users like you. Then it will be delivered to you in devices that do not allow you to change it.

This software might be “open source,” and use the open source development model; but it won’t be free software, since it won’t respect the freedom of the users that actually run it. If the open source development model succeeds in making this software more powerful and reliable for restricting you, that will make it even worse.

### **Fear of Freedom**

The main initial motivation for the term

## **Software can only be said to serve its users if it respects their freedom.**

“open source software” is that the ethical ideas of “free software” make some people uneasy. That’s true: talking about freedom, about ethical issues, about responsibilities as well as convenience, is asking people to think about things they might prefer to ignore, such as whether their conduct is ethical. This can trigger discomfort, and some people may simply close their minds to it. It does not follow that we ought to stop talking about these things.

However, that is what the leaders of “open source” decided to do. They figured that by keeping quiet about ethics and freedom, and talking only about the immediate practical benefits of certain free software, they might be able to “sell” the software more effectively to certain users, especially businesses.

This approach has proved effective, in its own terms. The rhetoric of open source has convinced many businesses and individuals to use, and even develop, free software, which has extended our community—but only at the superficial, practical level. The philosophy of open source, with its purely practical values, impedes understanding of the deeper ideas of free software; it brings many people into our community, but does not teach them to defend it. That is good, as far as it goes, but it is not enough to make freedom secure. Attracting users to free software takes them just part of the way to becoming defenders of their own freedom.

Sooner or later these users will be invited to switch back to proprietary software for some practical advantage. Countless companies seek to offer such temptation, some even offering copies gratis. Why would users decline? Only if they have learned to value the freedom free software gives them, to value freedom as such rather than the technical and practical convenience of specific free software. To spread this idea, we

have to talk about freedom. A certain amount of the “keep quiet” approach to business can be useful for the community, but it is dangerous if it becomes so common that the love of freedom comes to seem like an eccentricity.

That dangerous situation is exactly what we have. Most people involved with free software say little about freedom—usually because they seek to be more acceptable to business. Software distributors especially show this pattern. Nearly all GNU/Linux operating system distributions add proprietary packages to the basic free system, and they invite users to consider this an advantage, rather than a step backward from freedom.

Proprietary add-on software and partially non-free GNU/Linux distributions find fertile ground because most of our community does not insist on freedom with its software. This is no coincidence. Most GNU/Linux users were introduced to the system by “open source” discussion, which doesn’t say freedom is a goal. The practices that don’t uphold freedom and the words that don’t talk about freedom go hand in hand, each promoting the other. To overcome this tendency, we need more, not less, talk about freedom.

### **Conclusion**

As the advocates of open source draw new users into our community, we free software activists must work even more to bring the issue of freedom to those new users’ attention. We have to say, “It’s free software and it gives you freedom!”—more and louder than ever. Every time you say “free software” rather than “open source,” you help our campaign. □

### **Further Reading**

1. Joe Barr wrote an article called “Live and Let License” ([see <http://www.itworld.com/LWD010523control4>](http://www.itworld.com/LWD010523control4)) that gives his perspective on this issue.
2. Lakhani and Wolf’s paper on the motivation of free software developers ([see <http://freesoftware.mit.edu/papers/lakhaniwolf.pdf>](http://freesoftware.mit.edu/papers/lakhaniwolf.pdf)) states that a considerable fraction are motivated by the view that software should be free. This was despite the fact they surveyed the developers on SourceForge, a site that does not support the view that this is an ethical issue.

**Richard Stallman** ([rms@gnu.org](mailto:rms@gnu.org)) is the author of the free symbolic debugger GDB, the founder the project to develop the free GNU operating system, and the founder of the Free Software Foundation.

Copyright © 2009 Richard Stallman  
Verbatim copying and distribution of this entire article is permitted in any medium, provided this notice is preserved.



DOI:10.1145/1516046.1516057

George V. Neville-Neil

# Kode Vicious

## Obvious Truths

*How to determine when to put the brakes on late-running projects and untested software patches.*

### Dear KV,

I've been working on a project that, like all software projects, is late. And we're not just late a little, but a lot. The project was supposed to take four weeks and we're now in our third month. People are blaming the usual suspects: poorly spec'd-out work, management interference, and lack of proper infrastructure. What I want to know is how late is too late? How do people decide to just stop a project?

### Late and Getting Later

### Dear Later,

If I understand you correctly, and I hope I can because your email message is both short and direct, you are involved in a project that has now taken more than twice as long as predicted to implement and is approaching the thrice mark. I would say this is scary if it weren't so common. Projects take on a life of their own at some point and when a group of people get together and continue to try to "look on the bright side" they keep finding "silver linings," even though they are now drenched by the rain. It is amazing to me that a group of people who often seem so hard-headed and pragmatic—that is, engineers—can continue to believe there is a pot of gold just over the rainbow somewhere. Many projects can go on for years when they should have only gone on for months, so long as the money doesn't run out.

From my point of view there are a few good places to pause and reflect in the life of the project.

1. You have gotten to the end of the

originally published schedule and the work is not even 50% complete.

2. The originally published schedule has been extended by 50% or more.
3. The schedule is updated daily and the dates keep getting further out.
4. The engineers avoid coming to team meetings and when they do attend they:
  - a. break down in tears;
  - b. pretend to be asleep;
  - c. bang their heads on the table.

and the worse their output becomes. Pilots, fire fighters, emergency-room doctors, and the like all know that past a certain point everything they do will actually cause more trouble than if they did nothing at all. Because our profession is not as extreme as theirs we seem to never learn this, which is a shame, because it's an important lesson. Learn when to stop.

**KV**

### Driver Education

In this month's installment of "things that ought to be obvious" I discuss patching, compiling, and testing code. I'm sure many of you have had these experiences before, and if you have a fun one to share please write to me and tell me about it.

I am sure most of you have heard the old programmer's joke, "It compiles, ship it!" which gets a good guffaw now and again from the denizens of cubicle land. I'm also sure that many readers have been subjected to using code that clearly compiled, ran maybe once, and then actually was shipped. But have you ever had to deal with people sending you patches that just didn't work?

Recently, KV has been fixing a device driver that seems always to be very close to working. The driver wasn't originally written by KV, and it certainly wasn't originally tested by KV, although it now seems that the company I'm dealing with is using me as their unwitting alpha tester. There are few things more frustrating than a piece of software that almost works. It might tick along for days, doing just what it's



KV is in category C, but then I bet you knew that already. All of the above are indicators of schedule creep and a loss of control of the project. They are all good times to consider pulling the emergency brake handle. The reason the handle gets pulled so rarely is the aforementioned optimism of the staff, whereby if they "just work a little harder" the project will get done. I have never, in my entire working career, seen a project that is 50% off course get back on track because the team worked 80 hours a week instead of 60. Most people in high-pressure professions know how this works. The harder they work past a certain point, the more mistakes they make

supposed to when—bang!—it breaks. With a bit of debugging and a bit of time in the lab I can explain what's broken to the vendor. I even have source code for the driver so I can patch it when I understand what's broken and send them patches; sometimes they send me patches.

It's the part where they send me patches that has been a bit more interesting. I had been faithfully applying patches from the vendor and testing their fixes and I kept getting this sneaking feeling that they were not testing the patches before they sent them out. I had that feeling not just because I'm a naturally paranoid and suspicious person, which I am, but because each patch would fix say, only 70% or 80% of the problem and then I'd have to provide the remaining bit of the fix. Finally, I got a patch that proved that although I am paranoid, it is not without reason. I applied a patch and it didn't compile: the C keyword `struct` had been spelled incorrectly. Ha! I had them. They had not even applied their own patch; they were just

## People who send out a “small patch” without even compiling it are far too confident in their own abilities.

sending me hacked bits of the driver that they thought would work. All I could think was, “Did you even compile this!?!?” But of course I already knew the answer.

Now I don't bring this up just because I like to say, “I told you so,” because I don't. I'd much prefer the code I received worked the first time, since my employers expect the same from me. I bring this up as yet another example of unwarranted programmer optimism.

People who send out a “small patch” without even compiling it are far too confident in their own abilities. Please! Stop! Don't do that! I don't care if you see bits in your dreams and they assemble correctly in the morning when you type them in. The amount of time you waste by not doing the most basic tests on code you're patching isn't only your own; it's multiplied by all the hapless suckers who took your patch and tried to use it.

Returning to my earlier remark, I would have thought this was obvious—as obvious as how to spell “struct”—but I have discovered this is not the case.

**KV**

**George V. Neville-Neil** ([kv@acm.org](mailto:kv@acm.org)) is the proprietor of Neville-Neil Consulting and a member of the ACM Queue editorial board. He works on networking and operating systems code for fun and profit, teaches courses on various programming-related subjects, and encourages your comments, quips, and code snips pertaining to his *Communications* column.

Copyright held by author.

# HANDBOOK OF Statistical Analysis & Data Mining Applications

The *Handbook of Statistical Analysis and Data Mining Applications* is a comprehensive professional reference book for business analysts, scientists, engineers and researchers that brings together in a single resource all the information a beginner will need to rapidly learn how to conduct data mining and the statistical analysis required to interpret the data patterns once mined.

## key features:

- Provides key statistical analysis methods
- Clearly describes modern algorithms for AI/Machine learning
- Practical advice from successful real-world implementations
- Includes extensive case studies, examples, tutorials, MS PowerPoint slides and datasets

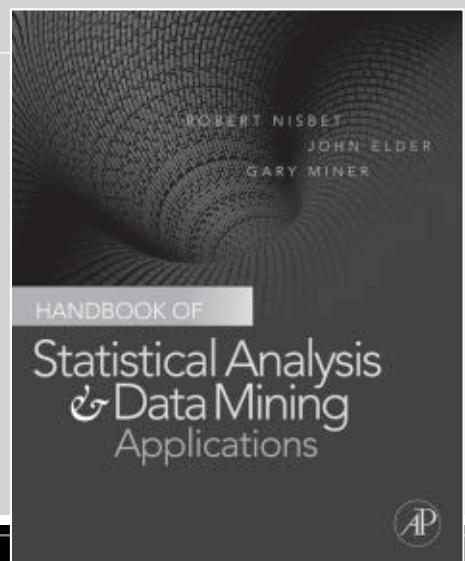
### authors:

**Robert Nisbet, PhD**  
Pacific Capital Bank Corporation  
Santa Barbara, CA, USA  
  
**John Elder, IV, PhD**  
Elder Research, Inc.  
Charlottesville, VA, USA  
  
**Gary Miner, PhD**  
StatSoft, Inc.  
Tulsa, OK, USA

*“If you want to roll-up your sleeves and execute on predictive analytics, this is your definite, go-to resource. To put it lightly, if this book isn't on your shelf, you're not a data miner.”*

— Eric Siegel, Ph.D.,  
President, Prediction Impact, Inc., San Francisco,  
and Founding Chair, Predictive Analytics World

ORDER TODAY & SAVE 15%



May 2009, Hardcover, 900 pp  
ISBN-13: 978-0-12-374765-5  
List Price: \$89.95/£45.99 / €57.95  
Use offer code 94637 when ordering.



>> To view the full Table of Contents or to order your copy, visit [elsevierdirect.com](http://elsevierdirect.com)

# **ACM, Uniting the World's Computing Professionals, Researchers, Educators, and Students**



Dear Colleague,

At a time when computing is at the center of the growing demand for technology jobs worldwide, ACM is continuing its work on initiatives to help computing professionals stay competitive in the global community. ACM's increasing involvement in activities aimed at ensuring the health of the computing discipline and profession serve to help ACM reach its full potential as a global and diverse society which continues to serve new and unique opportunities for its members.

As part of ACM's overall mission to advance computing as a science and a profession, our invaluable member benefits are designed to help you achieve success by providing you with the resources you need to advance your career and stay at the forefront of the latest technologies.

I would also like to take this opportunity to mention ACM-W, the membership group within ACM. ACM-W's purpose is to elevate the issue of gender diversity within the association and the broader computing community. You can join the ACM-W email distribution list at <http://women.acm.org/joinlist>.

## **ACM MEMBER BENEFITS:**

- A subscription to ACM's newly redesigned monthly magazine, **Communications of the ACM**
- Access to ACM's **Career & Job Center** offering a host of exclusive career-enhancing benefits
- **Free e-mentoring services** provided by MentorNet®
- **Full access to over 2,500 online courses** in multiple languages, and 1,000 virtual labs
- **Full access to 600 online books** from Safari® Books Online, featuring leading publishers, including O'Reilly (Professional Members only)
- **Full access to 500 online books** from Books24x7®
- Full access to the new **acmqueue** website featuring blogs, online discussions and debates, plus multimedia content
- The option to subscribe to the complete **ACM Digital Library**
- The **Guide to Computing Literature**, with over one million searchable bibliographic citations
- The option to connect with the **best thinkers in computing** by joining **34 Special Interest Groups or hundreds of local chapters**
- **ACM's 40+ journals and magazines** at special member-only rates
- **TechNews**, ACM's tri-weekly email digest delivering stories on the latest IT news
- **CareerNews**, ACM's bi-monthly email digest providing career-related topics
- **MemberNet**, ACM's e-newsletter, covering ACM people and activities
- **Email forwarding service & filtering service**, providing members with a free acm.org email address and **Postini** spam filtering
- And much, much more

ACM's worldwide network of over 92,000 members range from students to seasoned professionals and includes many of the leaders in the field. ACM members get access to this network and the advantages that come from their expertise to keep you at the forefront of the technology world.

Please take a moment to consider the value of an ACM membership for your career and your future in the dynamic computing profession.

Sincerely,

Wendy Hall



President

Association for Computing Machinery



Association for  
Computing Machinery



Association for  
Computing Machinery

*Advancing Computing as a Science & Profession*

# membership application & *digital library order form*

Priority Code: ACACM10

You can join ACM in several easy ways:

Online  
<http://www.acm.org/join>

Phone  
+1-800-342-6626 (US & Canada)  
+1-212-626-0500 (Global)

Fax  
+1-212-944-1318

Or, complete this application and return with payment via postal mail

**Special rates for residents of developing countries:**  
<http://www.acm.org/membership/L2-3/>

**Special rates for members of sister societies:**  
<http://www.acm.org/membership/dues.html>

*Please print clearly*

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_

State/Province \_\_\_\_\_

Postal code/Zip \_\_\_\_\_

Country \_\_\_\_\_

E-mail address \_\_\_\_\_

Area code & Daytime phone \_\_\_\_\_

Fax \_\_\_\_\_

Member number, if applicable \_\_\_\_\_

## Purposes of ACM

ACM is dedicated to:

- 1) advancing the art, science, engineering, and application of information technology
- 2) fostering the open interchange of information to serve both professionals and the public
- 3) promoting the highest professional and ethics standards

*I agree with the Purposes of ACM:*

*Signature* \_\_\_\_\_

ACM Code of Ethics:

<http://www.acm.org/serving/ethics.html>

## choose one membership option:

### PROFESSIONAL MEMBERSHIP:

- ACM Professional Membership: \$99 USD
- ACM Professional Membership plus the ACM Digital Library:  
\$198 USD (\$99 dues + \$99 DL)
- ACM Digital Library: \$99 USD (must be an ACM member)

### STUDENT MEMBERSHIP:

- ACM Student Membership: \$19 USD
- ACM Student Membership plus the ACM Digital Library: \$42 USD
- ACM Student Membership PLUS Print CACM Magazine: \$42 USD
- ACM Student Membership w/Digital Library PLUS Print  
CACM Magazine: \$62 USD

All new ACM members will receive an  
ACM membership card.  
For more information, please visit us at [www.acm.org](http://www.acm.org)

Professional membership dues include \$40 toward a subscription to *Communications of the ACM*. Member dues, subscriptions, and optional contributions are tax-deductible under certain circumstances. Please consult with your tax advisor.

### RETURN COMPLETED APPLICATION TO:

Association for Computing Machinery, Inc.  
General Post Office  
P.O. Box 30777  
New York, NY 10087-0777

Questions? E-mail us at [acmhelp@acm.org](mailto:acmhelp@acm.org)  
Or call +1-800-342-6626 to speak to a live representative

**Satisfaction Guaranteed!**

## payment:

Payment must accompany application. If paying by check or money order, make payable to ACM, Inc. in US dollars or foreign currency at current exchange rate.

Visa/MasterCard     American Express     Check/money order

Professional Member Dues (\$99 or \$198)    \$ \_\_\_\_\_

ACM Digital Library (\$99)    \$ \_\_\_\_\_

Student Member Dues (\$19, \$42, or \$62)    \$ \_\_\_\_\_

**Total Amount Due**    \$ \_\_\_\_\_

Card # \_\_\_\_\_

Expiration date \_\_\_\_\_

Signature \_\_\_\_\_

---

**New drive technologies and increased capacities create new categories of failure modes that will influence system designs.**

BY JON ELERATH

# Hard-Disk Drives: The Good, the Bad, and the Ugly

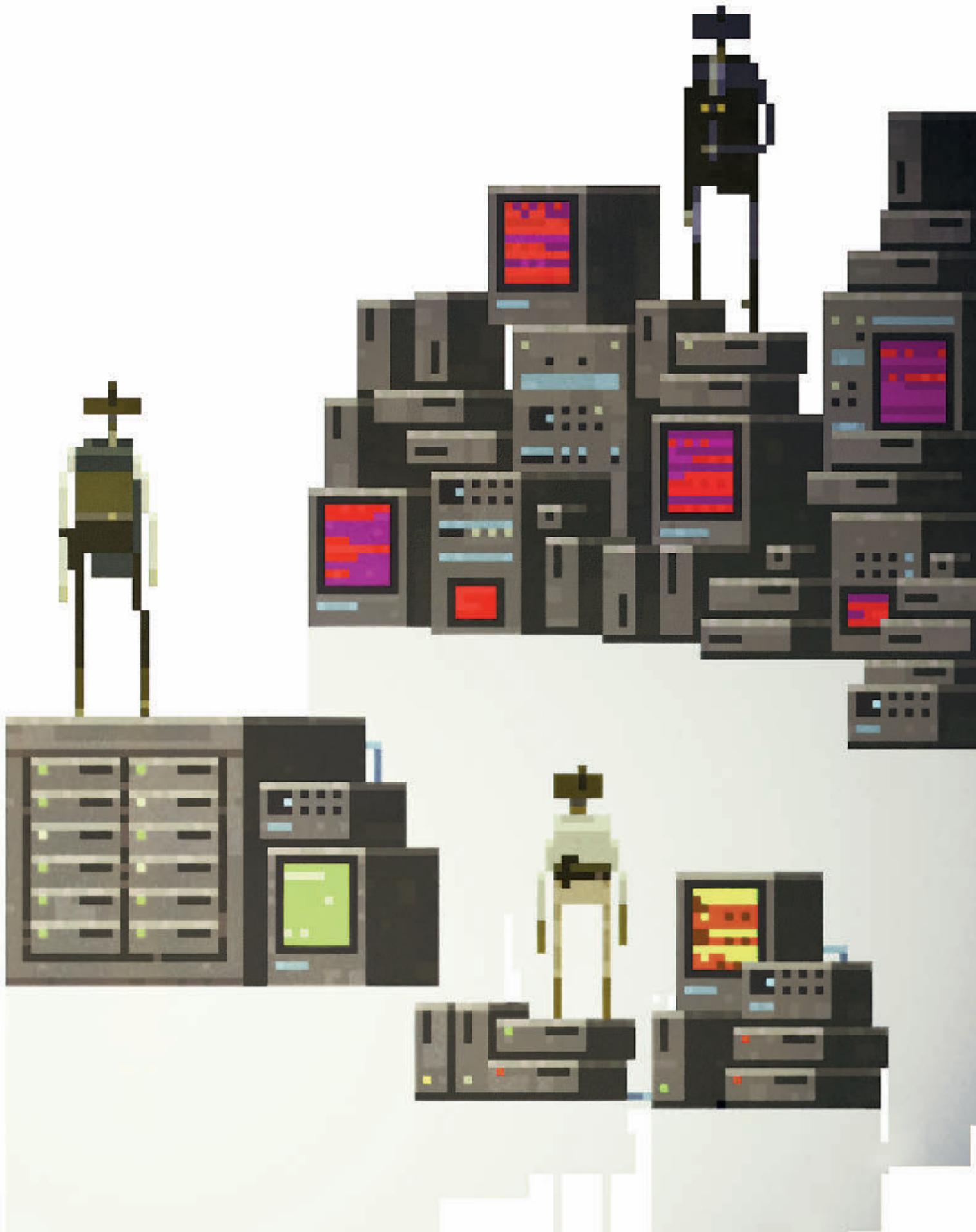
HARD-DISK DRIVES (HDDs) are like the bread in a peanut butter and jelly sandwich—seemingly unexciting pieces of hardware necessary to hold the software. They are simply a means to an end. HDD reliability, however, has always been a significant weak link, perhaps *the* weak link, in data storage. In the late 1980s people recognized that HDD reliability was inadequate for large data storage systems so redundancy was added at the system level with some brilliant software algorithms, and RAID (redundant array of independent disks) became a reality. RAID moved the reliability requirements from the HDD itself to the system of data disks. Commercial implementations of RAID include  $n+1$  configurations

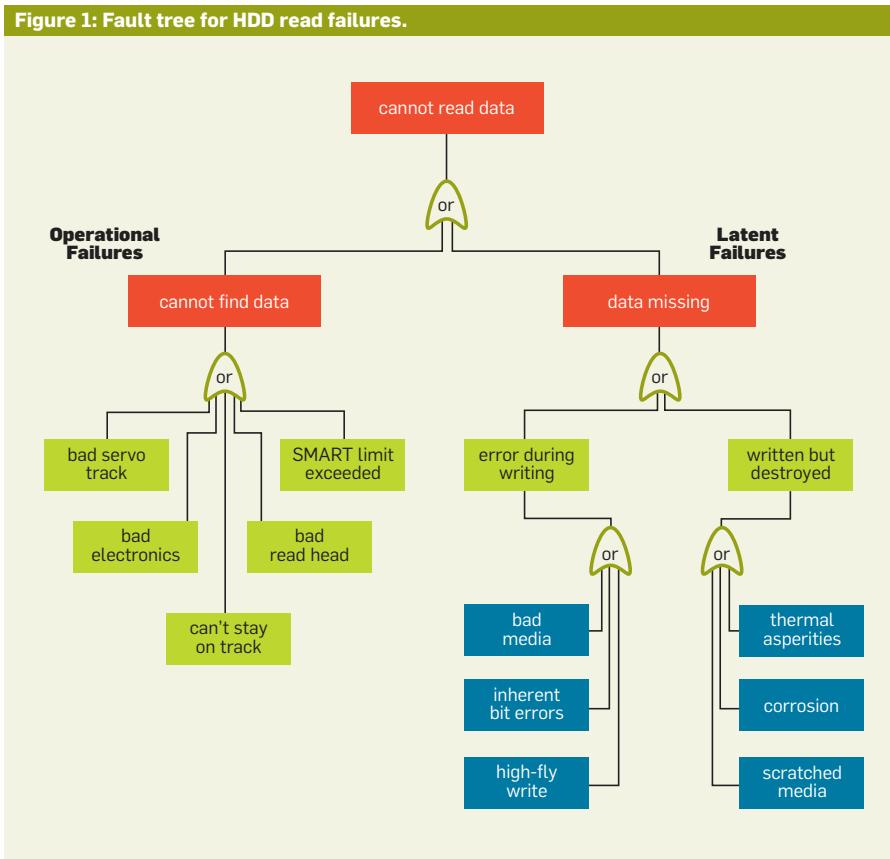
such as mirroring, RAID-4 and RAID-5, and the  $n+2$  configuration, RAID-6, which increases storage system reliability using two redundant disks (dual parity). Additionally, reliability at the RAID group level has been favorably enhanced because HDD reliability has been improving as well.

Several manufacturers produce one-terabyte HDDs and higher capacities are being designed. With higher areal densities (also known as bit densities), lower fly-heights (the distance between the head and the disk media), and perpendicular magnetic recording technology, can HDD reliability continue to improve? The new technology required to achieve these capacities is not without concern. Are the failure mechanisms or the probability of failure any different from predecessors? Not only are there new issues to address stemming from the new technologies, but also failure mechanisms and modes vary by manufacturer, capacity, interface, and production lot.

How will these new failure modes affect system designs? Understanding failure causes and modes for HDDs using technology of the current era and the near future will highlight the need for design alternatives and trade-offs that are critical to future storage systems. Software developers and RAID architects can not only better understand the effects of their decisions, but also know which HDD failures are outside their control and which they can manage, albeit with possible adverse performance or availability consequences. Based on technology and design, where must the developers and architects place the efforts for resiliency?

This article identifies significant HDD failure modes and mechanisms, their effects and causes, and relates them to system operation. Many failure mechanisms for new HDDs remain unchanged from the past, but the insidious undiscovered data corruptions (latent defects) that have plagued all HDD designs to one degree or another will continue to worsen in the near future as areal densities increase.



**Figure 1: Fault tree for HDD read failures.**

Two major categories of HDD failure can prevent access to data: those that fail the entire HDD and those that leave the HDD functioning but corrupt the data. Each of these modes has significantly different causes, probabilities, and effects. The first type of failure, which I term *operational*, is rather easy to detect, but has lower rates of occurrence than the data corruptions or *latent* defects that are not discovered until data is read. Figure 1 is a fault tree for the inability to read data—the topmost event in the tree—showing the two basic reasons that data cannot be read.

### **Operational Failures: Cannot Find Data**

Operational failures occur in two ways: first, data cannot be written to the HDD; second, after data is written correctly and is still present on the HDD uncorrupted, electronic or mechanical malfunction prevents it from being retrieved.

*Bad servo track.* Servo data is written at regular intervals on every data track of every disk surface. The servo data is used to control the positioning of the read/write heads. Servo data is

required for the heads to find and stay on a track, whether executing a read, write, or seek command. Servo-track information is written only during the manufacturing process and can be neither reconstructed using RAID nor rewritten in the field. Media defects in the servo-wedges cause the HDD to lose track of the heads' locations or where to move the head for the next read or write. Faulty servo tracks result in the inability to access data, even though the data is written and uncorrupted. Particles, contaminants, scratches, or thermal asperities can damage servo data.

*Can't stay on track.* Tracks on an HDD are not perfectly circular; some are actually spiral. The head position is continuously measured and compared with where it should be. A PES (position error signal) repositions the head over the track. This repeatable run-out is all part of normal HDD head positioning control. NRRO (nonrepeatable run-out) cannot be corrected by the HDD firmware since it is nonrepeatable. Caused by mechanical tolerances from the motor bearings, actuator arm bearings, noise, vibration, and servo-loop response errors, NRRO can

make the head positioning take too long to lock onto a track and ultimately produce an error. This mode can be induced by excessive wear and is exacerbated by high rotational speeds. It affects both ball and fluid-dynamic bearings. The insidious aspect of this type of problem is that it can be intermittent. Specific HDD usage conditions may cause a failure while reading data in a system, but under test conditions the problem might not recur.

Two very interesting examples of inability to stay on track are caused by audible noise. A video file available on YouTube shows a member of Sun's Fishworks team yelling at his disk drives and monitoring the latency in disk operations.<sup>5</sup> The vibrations from his yelling induce sufficient NRRO that the actuator cannot settle for over 520 ms. While most (some) of us don't yell at our HDDs, vibrations induced by thermal alarms (warning buzzers) have also been noted to induce NRRO and cause excessive latency and time-outs.

*SMART limits exceeded.* Today's HDDs collect and analyze functional and performance data to predict impending failure using SMART (self-monitoring analysis reporting technology). In general, sector reallocations are expected, and many spare sectors are available on each HDD. If an excessive number occurs in a specific time interval, however, the HDD is deemed unreliable and is failed out.

SMART isn't really that smart. One trade-off that HDD manufacturers face during design is the amount of RAM available for storing SMART data and the frequency and method for calculating SMART parameters. When the RAM containing SMART data becomes full, is it purged, then refilled with new data? Or are the most recent percentages ( $x\%$ ) of data preserved and the oldest ( $1-x\%$ )% purged? The former method means that a rate calculation such as read-error-rate can be erroneous if the memory fills up during an event that produces many errors. The errors before filling RAM may not be sufficient to trigger a SMART event, nor may the errors after the purge, but had the purge not occurred, the error conditions may easily have resulted in a SMART trip.

In general, the SMART thresholds are set very low, missing numerous

conditions that could proactively fail a HDD. Making the trip levels more sensitive (trip at lower levels) runs the risk of failing HDDs with a few errors that really aren't progressing to the point of failure. The HDD may simply have had a series of reallocations, say, that went smoothly, mapping out the problematic area of the HDD. Integrators must assess the HDD manufacturer's implementation of SMART and see if there are other more instructive calculations. Integrators must at least understand the SMART data collection and analysis process at a very low level, then assess their specific usage pattern to decide whether the implementation of SMART is adequate or whether the SMART decisions need to be moved up to the system (RAID group) level.

*Head games and electronics.* Most head failures result from changes in the magnetic properties, not electrical characteristics. ESD (electrostatic discharge), high temperatures, and physical impact from particles affect magnetic properties. As with any highly integrated circuit, ESD can leave the read heads in a degraded mode. Subsequent moderate to low levels of heat may be sufficient to fail the read heads magnetically. A recent publication from Google didn't find a significant correlation between temperature and reliability.<sup>6</sup> In my conversations with numerous engineers from all the major HDD manufacturers, none has said the temperature does not affect head reliability, but none has published a transfer function relating head life to time and temperature. The read element is physically hidden and difficult to damage, but heat can be conducted from the shields to the read element, affecting magnetic properties of the reader element, especially if it is already weakened by ESD.

The electronics on an HDD are complex. Failed DRAM and cracked chip capacitors have been known to cause HDD failure. As the HDD capacities increase, the buffer sizes increase and more RAM is required to cache writes. Is RAID at the RAM level required to assure reliability of the ever-increasing solid-state memory?

## Operational Failure Data

In a number of studies on disk failure rates, all mean times between fail-

ures disagree with the manufacturers' specification.<sup>1–3, 6, 7, 10, 11</sup> More disconcerting are the realizations that the failure rates are rarely constant; there are significant differences across suppliers, and great differences within a specific HDD family from a single supplier. These inconsistencies are further complicated by unexpected and uncontrolled lot-to-lot differences.

In a population of HDDs that are all the same model from a single manufacturer, there can be statistically significant subpopulations, each having a different time-to-failure distribution with different parameters. Analyses of HDD data indicate these subpopulations are so different that they should not be grouped together for analyses because the failure causes and modes are different. HDDs are a technology that defies the idea of "average" failure rate or MTBF; inconsistency is synonymous with variability and unpredictability.

The following are examples of unpredictability that existed to such an extent that at some point in the product's life, these subpopulations dominated the failure rate:

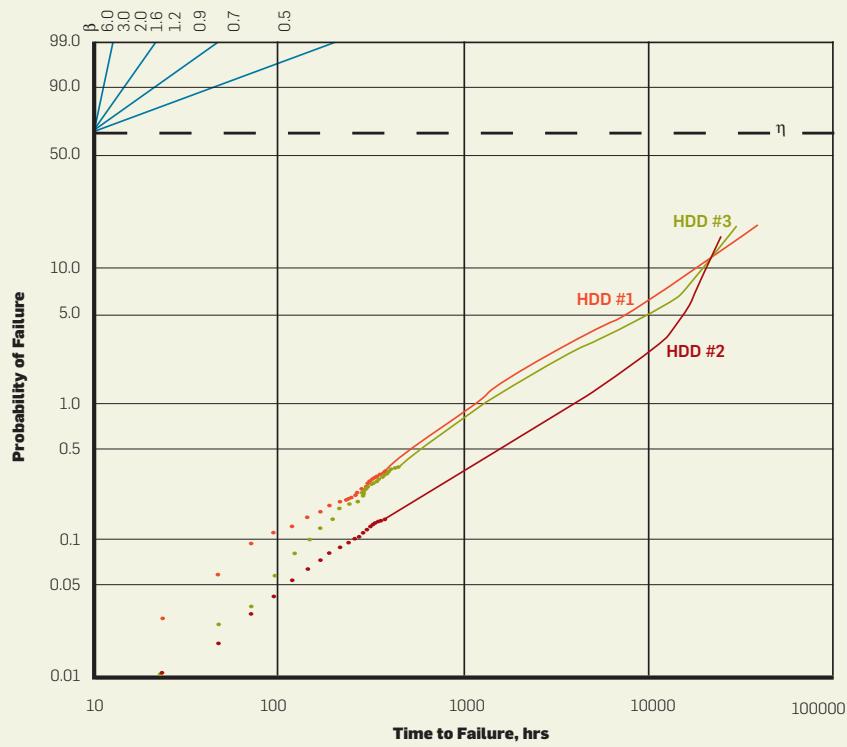
- Airborne contamination. Particles within the enclosure tend to fail HDDs early (scratches and head damage). This can give the appearance of an increasing failure rate. After all the contaminated HDDs fail, the failure rate often decreases.

- Design changes. Manufacturers periodically find it necessary to reduce cost, resolve a design issue discovered late in the test phase, or improve yields. Often, the change creates an improvement in field reliability, but can create more problems than it solves. For example, one design change had an immediately positive effect on reliability, but after two years another failure mode began to dominate and the HDD reliability became significantly worse.

- Yield changes. HDD manufacturers are constantly tweaking their processes to improve yield. Unfortunately, HDDs are so complex that these yield enhancements can inadvertently reduce reliability. Continuous tweaks can result in one month's production being highly reliable and another month being measurably worse.

The net impact of variability in reli-

**Figure 2: Weibull time to failure plot for three very different populations.**



ability is that RAID designers and software developers must develop logic and operating rules that will accommodate significant variability and the worst-case issues for all HDDs. Figure 2 shows a plot for three different HDD populations. If a straight line were to fit the data points and the slope were 1.0, then the population could be represented by a Weibull probability distribution and have a constant failure rate. (The Weibull distribution is used to create the common bathtub curve.) A single straight line cannot fit either population HDD#2 or HDD#3, so they do not even fit a Weibull distribution. In fact, these do not fit any single closed-form distribution, but are composed of multiple failure distributions from causes that dominate at different points in time. Figure 3 is an example of five HDD vintages from a single supplier. A straight line indicates a constant failure rate; the lower the slope, the more reliable the HDD. A vintage represents a product from a single month.

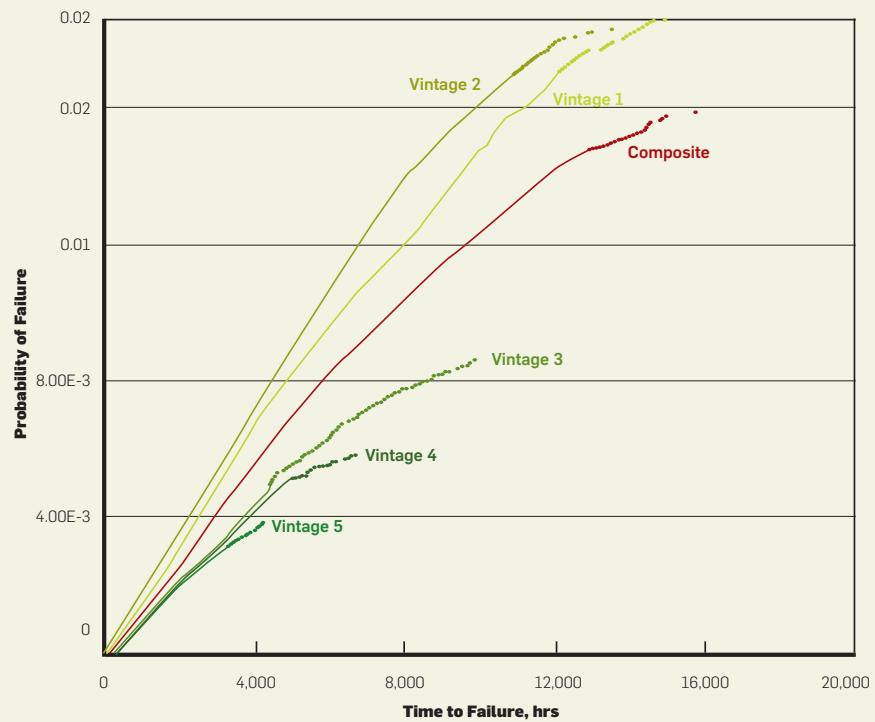
### Latent Defects: Data is Corrupted or Missing

The preceding discussion centered on failure modes in which data was good (uncorrupted) but some other electrical, mechanical, or magnetic function was impaired. These modes are usually rather easily detected and allow the system operator to replace the faulty HDD, reconstruct data on the new HDD, and resume storage functions. But what about data that is missing or corrupted because it either was not written well initially or was erased or corrupted *after* being written well. All errors resulting from missing data are latent because the corrupted data is resident without the knowledge of the user (software). The importance of latent defects cannot be overemphasized. The combination of a latent defect followed by an operational failure is the most likely sequence to result in a double failure and loss of data.<sup>1</sup>

To understand latent defects better, consider the common causes.

Write errors can be corrected using a read-verify command, but these require an extra read command after writing, and can nearly double the effective time to write data. The BER (bit-error rate) is a statistical measure

**Figure 3: Failure rate over time for five vintages and the composite.**



of the effectiveness of all the electrical, mechanical, magnetic, and firmware control systems working together to write (or read) data. Most bit errors occur on a read command and are corrected using the HDD's built-in error-correcting code algorithms, but errors can also occur during writes. While BER does account for some fraction of defective data, a greater source of data corruption is the magnetic recording media coating the disks.

The distance that the read-write head flies above the media is carefully controlled by the aerodynamic design of the slider, which contains the reader and writer elements. In today's designs, the fly height is less than 0.3  $\mu$ -in. Events that disturb the fly height, increasing it above the specified height during a write, can result in poorly written data because the magnetic-field strength is too weak. Remember that magnetic-field strength does not decrease linearly as a function of distance from the media, but is a power function, so field strength falls off very rapidly as the distance between the head and media increases. Writing data while

the head is too high can result in the media being insufficiently magnetized so it cannot be read even when the read element is flying at the specified height. If writing over a previously written track, the old data may persist where the head was flying too high. For example, if all the HDDs in a cabinet are furiously writing at the same time, self-induced vibrations and resonances can be great enough to affect the fly height. Physically bumping or banging an HDD during a write or walking heavily across a poorly supported raised floor can create excessive vibration that affects the write.

A more difficult problem to solve is persistent increase in the fly height caused by buildup of lubrication or other hydrocarbons on the surface of the slider. Hydrocarbon lubricants are used in three places within enclosed HDDs. To reduce the NRRO, motors often use fluid-dynamic bearings. The actuator arm that moves the heads pivots using an enclosed bearing cartridge that contains a lubricant. The media itself also has a very thin layer of lubricant applied to prevent the

heads from touching the media itself. Lubricants on the media can build up on the head under certain circumstances and cause the head to fly too high. Lube buildup can also mean that uncorrupted, well-written data cannot be read because the read element is too far from the media. Lube buildup can be caused by the mechanical properties of the lubricant, which is dependent upon the chemical composition. Persistent high fly height can also be caused by specific operations. For example, when not writing or reading, if the head is left to sit above the same track while the disks spin, lubricant can collect on the heads. In some cases simply powering down the HDD will cause the heads to touch down (as they are designed to do) in the landing zone to disturb the lube buildup. This is very design specific, however, and does not always work.

During the manufacturing process, the surface of the HDD is checked and defects are mapped out, and the HDD firmware knows not to write in these locations. They also add “padding” around the defective area mapping out more blocks than the estimated minimum, creating additional physical distance around the defect that is not available for storing data. Since it is difficult to determine the exact length, width, and shape of a defect, the added padding provides an extra safeguard against writing on a media defect.

Media imperfections such as voids (pits), scratches, hydrocarbon contamination (various oils), and smeared soft particles can not only cause errors during writing, but also corrupt data after it has been written. The sputtering process used to apply some of the media layers can leave contaminants buried within the media. Subsequent contact by the slider can remove these bumps, leaving voids in which the media is defective. If data is already written there, the data is corrupted. If none is written, the next write process will be unsuccessful, but the user won’t know this unless a write-verify command is used.

Early reliability analyses assumed that once written, data will remain undestroyed except by degradation of the magnetic properties of the media, a process known as bit-rot. Bit-rot, in

## Based on technology and design, where must the developers and architects place the efforts for resiliency?

which the magnetic media is not capable of holding the proper magnetic field to be correctly interpreted as a 0 or a 1, is really not an issue. Media can degrade, but the probability of this mode is inconsequential compared with other modes. Data can become corrupted whenever the disks are spinning, even when data is not being written to or read from the disk. Common causes for erasure include thermal asperities, corrosion, and scratches or smears.

Thermal asperities are instances of high heat for a short duration caused by head-disk contact. This is usually the result of heads hitting small “bumps” created by particles that remain embedded in the media surface even after burnishing and polishing. The heat generated on a single contact can be high enough to erase data. Even if not on the first contact, cumulative effects of numerous contacts may be sufficient to thermally erase data or mechanically destroy the media coatings and erase data.

The sliders are designed to push away airborne particles so they do not become trapped between the head and disk surface. Unfortunately, removing all particles that are in the 0.3  $\mu\text{-in.}$  range is very difficult, so particles do get caught. Hard particles used in the manufacture of an HDD, such as  $\text{Al}_2\text{O}_3$ , TiW, and C, will cause surface scratches and data erasure. These scratches are then media defects that are not mapped out, so the next time data is written to those locations the data will be corrupted immediately. Other “soft” materials such as stainless steel can come from assembly tooling and aluminum from residuals from machining the case. Soft particles tend to smear across the surface of the media rendering the data unreadable and unwritable. Corrosion, although carefully controlled, can also cause data erasure and may be accelerated by high ambient heat within the HDD enclosure and the very high heat flux from thermal asperities.

### Latent Defects Data

Latent defects are the most insidious kinds of errors. These data corruptions are present on the HDD but undiscovered until the data is read. If no operational failures occur at the first

reading of the data, the corruption is corrected using the parity disk and no data is lost. If one HDD, however, has experienced an operational failure and the RAID group is in the process of reconstruction when the latent defect is discovered, that data is lost. Since latent defects persist until discovered (read) and corrected, their rate of occurrence is an extremely important aspect of RAID reliability.

One study concludes that the BER is fairly inconsequential in terms of creating corrupted data,<sup>4</sup> while another claims the rate of data corruption is five times the rate of HDD operating failures.<sup>8</sup> Analyses of corrupted data identified by specific SCSI error codes and subsequent detailed failure analyses show that the rate of data corruption for all causes is significant and must be included in the reliability model.

NetApp (Network Appliance) completed a study in late 2004 on 282,000 HDDs used in RAID architecture. The RER (read-error rate) over three months was  $8 \times 10^{-14}$  errors per byte read. At the same time, another analysis of 66,800 HDDs showed an RER of approximately  $3.2 \times 10^{-13}$  errors per byte. A more recent analysis of 63,000 HDDs over five months showed a much-improved  $8 \times 10^{-15}$  errors per byte read. In these studies, data corruption is verified by the HDD manufacturer as an HDD problem and not a result of the operating system controlling the RAID group.

While Jim Gray of Microsoft Research asserted that it is reasonable to transfer  $4.32 \times 10^{12}$  bytes/day/HDD, the study of 63,000 HDDs read  $7.3 \times 10^{17}$  bytes of data in five months, an approximate read rate of  $2.7 \times 10^{11}$  bytes/day/HDD.<sup>4</sup> Using combinations of the

RERs and number of bytes read yields the hourly read failure rates shown in the table here.

Latent defects do not occur at a constant rate, but in bursts or adjacent physical (not logical) locations. Although some latent defects are created by wear-out mechanisms, data is not available to discern wear-out from those that occur randomly at a constant rate. These rates are between 2 and 100 times greater than the rates for operational failures.

### Potential Value of Data Scrubbing

Latent defects (data corruptions) can occur during almost any HDD activity: reading, writing, or simply spinning. If not corrected, these latent defects will result in lost data when an operational failure occurs. They can be eliminated, however, by background scrubbing, which is essentially preventive maintenance on data errors. During scrubbing, which occurs during times of idleness or low I/O activity, data is read and compared with the parity. If they are consistent, no action is taken. If they are inconsistent, the corrupted data is recovered and rewritten to the HDD. If the media is defective, the recovered data is written to new physical sectors on the HDD and the bad blocks are mapped out.

If scrubbing does not occur, the period of time to accumulate latent defects starts when the HDD begins operation in the system. Since scrubbing requires reading and writing data, it can act as a time-to-failure accelerator for HDD components with usage-dependent time-to-failure mechanisms. The optimal scrub pattern, rate, and time of scrubbing is HDD-specific and must be determined in conjunction with the HDD manufacturer to assure

that operational failure rates are not increased.

Frequent scrubbing can affect performance, but too infrequent scrubbing makes the  $n+1$  RAID group highly susceptible to double disk failures. Scrubbing, as with full HDD data reconstruction, has a minimum time to cover the entire HDD. The time to complete the scrub is a random variable that depends on HDD capacity and I/O activity. The operating system may invoke a maximum time to complete scrubbing.

### Future Technology and Trade-Offs

How are those failure modes going to impact future HDDs that have more than one-terabyte capacity? Certainly, all the failure mechanisms that occur in the 1TB drive will persist in higher density drives that use perpendicular magnetic recording (PMR) technology. PMR uses a "thick," somewhat soft underlayer making it susceptible to media scratching and gouging. The materials that cause media damage include softer metals and compositions that were not as great a problem in older, longitudinal magnetic recording. Future higher density drives are likely to be even more susceptible to scratching because the track width will be narrower.

Another PMR problem that will persist as density increases is side-track erasure. Changing the direction of the magnetic grains also changes the direction of the magnetic fields. PMR has a return field that is close to the adjacent tracks and can potentially erase data in those tracks. In general, the track spacing is wide enough to mitigate this mechanism, but if a particular track is written repeatedly, the probability of side-track erasure increases. Some applications are optimized for performance and keep the head in a static position (few tracks). This increases the chances of not only lube buildup (high fly writes) but also erasures.

One concept being developed to increase bit-density is heat assisted magnetic recording (HAMR).<sup>9</sup> This technology requires a laser within the write head to heat a very small area on the media to enable writing. High-stability media using iron-platinum alloys allow bits to be recorded on much

#### Range of average read error rates.

Read Errors per Byte per HDD	Bytes Read per Hour	
	Low rate ( $1.35 \times 10^9$ )	High rate ( $1.35 \times 10^{10}$ )
Low ( $8.0 \times 10^{-15}$ )	$1.08 \times 10^{-5}$ err/hr	$1.08 \times 10^{-4}$ err/hr
Medium ( $8.0 \times 10^{-14}$ )	$1.08 \times 10^{-4}$ err/hr	$1.08 \times 10^{-3}$ err/hr
High ( $3.2 \times 10^{-13}$ )	$4.32 \times 10^{-4}$ err/hr	$4.32 \times 10^{-3}$ err/hr

smaller areas than today's standard media without being limited by super-paramagnetism. Controlling the amount and location of the heat are, of course, significant concerns.

RAID is designed to accommodate corrupted data from scratches, smears, pits, and voids. The data is re-created from the parity disk and the corrupted data is reconstructed and rewritten. Depending on the size of the media defect, this may be a few blocks or hundreds of blocks. As the areal density of the HDDs increases, the same physical size of the defect will affect more blocks or tracks and require more time for re-creation of data. One trade-off is the amount of time spent recovering corrupted data. A desktop HDD (most ATA drives) is optimized to find the data no matter how long it takes. In a desktop there is no redundancy and it is (correctly) assumed that the user would rather wait 30–60 seconds and eventually retrieve the data than to have the HDD give up and lose data.

Each HDD manufacturer has a proprietary set of recovery algorithms it employs to recover data. If the data cannot be found, the servo controller will move the heads a little to one side of the nominal center of the track, then to the other side. This off-track reading may be performed several times at different off-track distances. This is a very common process used by all HDD manufacturers, but how long can a RAID group wait for this recovery?

Some RAID integrators may choose to truncate these steps with the knowledge that the HDD will be considered failed even though it is not an operational failure. On the other hand, how long can a RAID group response be delayed while one HDD is trying to recover data that is readily recoverable using RAID? Also consider what happens when a scratch is encountered. The process of recovery for a large number of blocks, even if the process is truncated, may result in a time-out condition. The HDD is off recovering data or the RAID group is reconstructing data for so long that the performance comes to a halt; a time-out threshold is exceeded and the HDD is considered failed.

One option is quickly to call the offending HDD failed, copy all the data

to a spare HDD (even the corrupted data), and resume recovery. A copy command is much quicker than reconstructing the data based on parity, and if there are no defects, little data will be corrupted. This means that reconstruction of this small amount of data will be fast and not result in the same time-out condition. The offending HDD can be (logically) taken out of the RAID group and undergo detailed diagnostics to restore the HDD and map out bad sectors.

In fact, a recent analysis shows the true impact of latent defects on the frequency of double disk failures.<sup>1</sup> Early RAID papers stated that the only failures of concern were operational failures because, once written, data does not change except by bit-rot.

### Improving Reliability

Hard-disk drives don't just fail catastrophically. They may also silently corrupt data. Unless checked or scrubbed, these data corruptions result in double disk failures if a catastrophic failure also occurs. Data loss resulting from these events is the dominant mode of failure for an  $n+1$  RAID group. If the reliability of RAID groups is to increase, or even keep up with technology, the effects of undiscovered data corruptions must be mitigated or eliminated. Although scrubbing is one clear answer, other creative methods to deal with latent defects should be explored.

Multi-terabyte capacity drives using perpendicular recording will be available soon, increasing the probability of both correctable and uncorrectable errors by virtue of the narrowed track widths, lower flying heads, and susceptibility to scratching by softer particle contaminants. One mitigation factor is to turn uncorrectable errors into correctable errors through greater error-correcting capability on the drive (4KB blocks rather than 512- or 520-byte blocks) and by using the complete set of recovery steps. These will decrease performance, so RAID architects must address this trade-off.

Operational failure rates are not constant. It is necessary to analyze field data, determine failure modes and mechanisms, and implement corrective actions for those that are most problematic. The operating system

should consider optimizations around these high-probability events and their effects on the RAID operation.

Only when these high-probability events are included in the optimization of the RAID operation will reliability improve. Failure to address them is a recipe for disaster. ■

### Related articles on queue.acm.org

#### You Don't Know Jack about Disks

Dave Anderson

<http://queue.acm.org/detail.cfm?id=864058>

#### CTO Roundtable: Storage

<http://queue.acm.org/detail.cfm?id=1466452>

#### A Conversation with Jim Gray

<http://queue.acm.org/detail.cfm?id=864078>

### References

1. Elerath, J.G. Reliability model and assessment of redundant arrays of inexpensive disks (RAID) incorporating latent defects and non-homogeneous poisson process events. Ph.D. dissertation, Department of Mechanical Engineering, University of Maryland, 2007.
2. Elerath, J.G. and Pecht, M. Enhanced reliability modeling of RAID storage systems. In *Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, (Edinburgh, UK, June 2007).
3. Elerath, J.G. and Shah, S. Server class disk drives: How reliable are they? In *Proceedings of the Annual Reliability and Maintainability Symposium*, (January 2004), 151–156.
4. Gray, J. and van Ingen, C. Empirical measurements of disk failure rates and error rates. Microsoft Research Technical Report, MSR-TR-2005-166, December 2005.
5. Gregg, B. Shouting in the datacenter, 2008; <http://www.youtube.com/watch?v=tDacjrSCeq4>.
6. Pinheiro, E., Weber, W.-D., and Barroso, L.A. Failure trends in a large disk drive population. In *Proceedings of the Fifth Usenix Conference on File and Storage Technologies (FAST)*, (February 2007).
7. Schroeder, B. and Gibson, G. Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you? In *Proceedings of the Fifth Usenix Conference on File and Storage Technologies (FAST)*, (February 2007).
8. Schwarz, T.J.E., et al. Disk scrubbing in large archival storage systems. In *Proceedings of the IEEE Computer Society Symposium* (2004), 1161–1170.
9. Seigler, M. and McDaniel, T. What challenges remain to achieve heat-assisted magnetic recording? *Solid State Technology* (Sept. 2007); [http://www.solid-state.com/display\\_article/304597/5/ARTCL/none/none/What-challenges-remain-to-achieve-heat-assisted-magnetic-recording/](http://www.solid-state.com/display_article/304597/5/ARTCL/none/none/What-challenges-remain-to-achieve-heat-assisted-magnetic-recording/).
10. Shah, S. and Elerath, J.G. Disk drive vintage and its affect on reliability. In *Proceedings of the Annual Reliability and Maintainability Symposium*, (January 2004), 163–167.
11. Sun, F. and Zhang, S. Does hard-disk drive failure rate enter steady-state after one year? In *Proceedings of The Annual Reliability and Maintainability Symposium*, IEEE, (January 2007).

**Jon Elerath** is a staff reliability engineer at SolFocus. He has focused on hard-disk drive reliability for more than half his 35-plus-year career, which includes positions at NetApp, General Electric, Tegal, Tandem Computers, Compaq, and IBM.

## The history of NFE processors sheds light on the trade-offs involved in designing network stack software.

BY MIKE O'DELL

# Network Front-end Processors, Yet Again

*“This time for sure, Rocky!”*

—Bullwinkle J. Moose

THE HISTORY OF the network front-end (NFE) processor, best known as a TCP offload engine (or TOE), extends back to the Arpanet interface message processor and possibly before. The notion is beguilingly simple: partition the work of executing communications protocols from the work of executing the applications that require the services of those protocols. That way, the applications and the network machinery can achieve maximum performance and efficiency, possibly taking advantage of special hardware performance assistance. While this looks utterly compelling on the whiteboard, architectural

and implementation realities intrude, often with considerable force.

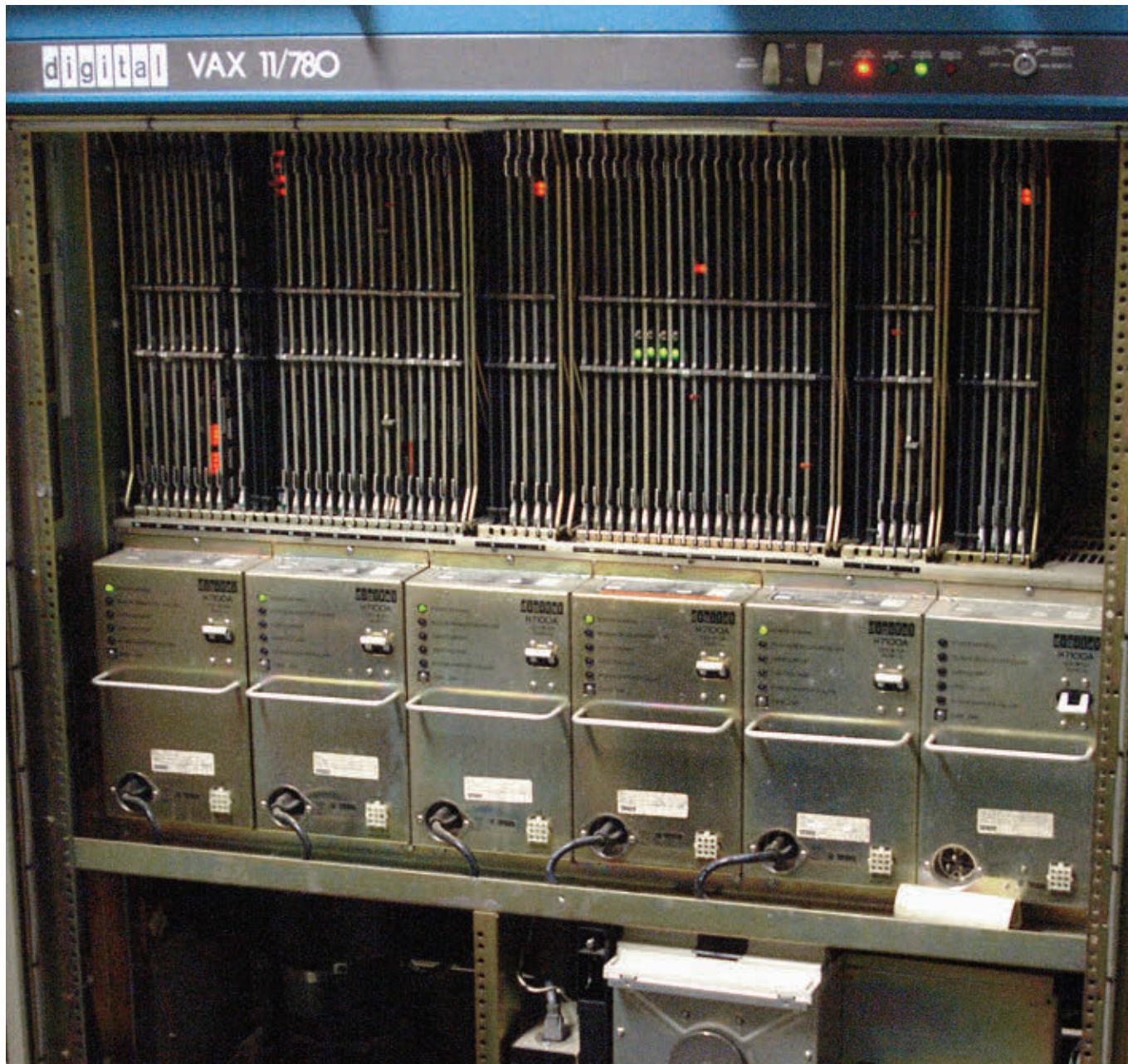
This article will not attempt to discern whether the NFE is a heavenly gift or a manifestation of evil incarnate. Rather, it will follow its evolution starting from a pure host-based implementation of a network stack and then moving the network stack farther from that initial position, observing the issues that arise. The goal is to offer insight into the trade-offs that influence the location choice for network stack software in a larger systems context. As such, it is an attempt to prevent old mistakes from being reinvented while harvesting as much clean grain as possible.

As a starting point, consider the canonical structure of a common workstation or server before the advent of multicore processors. Ignoring the provenance of the operating-system code, this model springs directly from the quintessential early to mid-1980s computer science department computer, the DEC VAX 11/780 with a 10Mb Ethernet interface with single-cycle direct memory access (DMA) ability and connected to a relatively slow 16-bit bus (the DEC Unibus).

Since there is only one processor, the network stack vies for the attention of the CPU with everything else running on the machine, albeit probably with the aid of a software priority mechanism that makes the network code “more equal than others.”

When a packet arrives, the Ethernet interface validates the Ethernet frame cyclic redundancy check (CRC) and then uses DMA to transfer the packet into buffers used by the network code for protocol processing. The DMA transfers require only one local bus cycle for each 16-bit word, and on the VAX 11/780 the processor controller for the Unibus buffers 16-bit words into a single 32-bit transfer into main memory.

The TCP checksum is then calculated by the network code, the protocol state machinery conducts its business, and the TCP payload data is copied into “socket buffers” to await consumption



A VAX-11/780 from 1983 with 16MB of RAM, and the Ethernet interface containing a Motorola 68000 processor to handle the network traffic.

by the application program. When the read for the payload data happens, it is copied from the socket buffer into application process memory to be digested as required. That makes a total of four passes over the data in a single packet before the application gets a shot at using it. When networks were slow compared with memory bandwidth and processor speed, the data-copy inefficiency was considered minor compared with the joy of a working network stack, so it failed to provoke immediate improvement.

This base-case platform appears to be the origin of the folk theorem that "TCP needs one (VAX-)MIPS per

10 megabits/second of network performance." The 10Mbps Ethernet can deliver about a megabyte/second of payload, so this is consistent with the other folk theorem of "one megabyte of memory per MIPS per megabyte of I/O." Where this came from is difficult to pin down, but it is frequently credited to Gene Amdahl.

Now, let's move this same model to PC hardware. For a long time, one of the principal distinctions between PCs and minicomputers was I/O performance. To be brutal, compared with its minicomputer forebears, the PC platform started life with almost no I/O capabilities. Over the life of the

PC platform, that conspicuous lack prompted major renovations of the PC's I/O architecture. For the period of our interest, that progressed from the 16-bit ISA bus, to 32-bit PCI, and now PCI Express. For reasons too boring to explore here, for a very long time, packets moved from PC Ethernet cards into protocol processing buffers with a byte-copy operation performed by the CPU, upping the data-handling pass count to five.

The first significant improvement came when the raw-packet copy operation and TCP checksum were combined. Some network code tried to do this in software. As PCI Ethernet

cards developed efficient DMA hardware, some combined the TCP checksum generation with the copy operation, reducing the pass count to three. This clearly reduced CPU use for a given amount of TCP throughput and started the march to “protocol assist” services performed by network interfaces. (“If a little help is good, a lot of help should be better!”) Adapting the network stack code to exploit this new checksum capability was not trivial, but the handwriting on the wall made it clear that such evolution was likely to continue. Significant redesign of the network code had to be done to allow functions to move between hardware and software with greater ease in the future. This was genuine architectural progress, although it did not happen overnight.

#### A Success Disaster

With the explosion of the Web, performance demands on network servers skyrocketed. Processors and network interfaces were getting faster, and memory bandwidth strangulation was being solved. Gigabit Ethernet quickly became commonplace on server motherboards (and gamer desktop motherboards!). By this time, the cost of all those data copies was clearly unacceptable. Simply halving the number of copies would come close to doubling the sustainable transaction rate for many Web workloads.

This gave rise to the Holy Grail of what became known as *zero-copy TCP*. The idea was that programs written to exploit this new capability could have data delivered right into application buffers without any intervening copies (ignoring the possible exception of one efficient DMA transfer from the hardware). Clearly this would require some cooperation (or at least reduced antagonism) from designers of Ethernet interface hardware, but a working solution would win many hearts and minds.

The step from a zero-copy TCP network stack to a full-blown TCP offload engine looks pretty obvious at this point. It seems even more attractive given that many PC-based platforms were slow to exploit the multiprocessor abilities the PC was developing. (Whether it is multiple chips or multiple cores on one chip is largely irrelevant.) The

**Simply moving data directly off the network wire into application buffers is not sufficient. The delivery of packets must be coordinated with all the other things the application is doing and all the other operating-system machinery behind the scenes.**

ability to add a fast processor that can be applied entirely to protocol processing is certainly an attractive idea. It is, however, much more difficult to do in real life than it first appears on the whiteboard.

Simply moving data directly off the network wire into application buffers is not sufficient. The delivery of packets must be coordinated with all the other things the application is doing and all the other operating-system machinery behind the scenes. As a result, the network protocol stack interacts with the rest of the operating system in exquisitely delicate ways. Truth be told, this coordination machinery is the lion’s share of the code in most stack implementations. The actual TCP state machine fits on a half page, once divorced of all the glue and scaffolding needed to integrate it with the rest of the system environment. It is precisely this subtle and complex control coupling that makes it surprisingly difficult to isolate a network protocol stack fully from its host operating system. There are multiple reasons why this interaction is such a rich breeding ground for implementation bugs, but one vast category is “abstraction mismatch.”

Because communications protocols inherently deal with multiple communicating entities, some assumptions must be made about the behavior of those entities. The degree to which those assumptions match between a host system and protocol code determines how difficult it will be to map to existing semantics and how much new structure and machinery will be required. When networking first went into Berkeley Unix, subtleties on both sides required considerable effort to reconcile. There was a critical desire to make network connections appear to be natural extensions of existing Unix machinery: file descriptors, pipes, and the other ideas that make Unix conceptually compact. But because of radical differences in behavior, especially delay, it is impossible to completely disguise reading 1,000 bytes from a round-the-world network connection so that it appears indistinguishable from reading that same 1,000 bytes from a file on a local file system. Networks have new behaviors that require new interfaces to capture and manage, but those new interfaces must make sense with exist-

ing interfaces. This was difficult work, and the modifications left few pieces of the system untouched; a few changed in profound ways.

The fundamental capabilities provided by a network protocol stack are data transfer, multiplexing, flow control, and error management. All of these functions are required for the coordinated delivery of data between endpoints across the Internet. Indeed, the purpose of all the structure in the packet headers: to carry the control coordination information, as well as the payload data.

The critical observation is that the exact same operations are required to coordinate the interaction of a network protocol stack and the host operating system within a single system. When all the code is in the same place (that is, running on the same processor), this signaling is easily done with simple procedure calls. If, however, the network protocol stack executes on a remote processor such as a TOE, this signaling must be done with an explicit protocol carried across whatever connects the front-end processor to the host operating system. This protocol is called a host-front end protocol (HFEP).

Designing an HFEP is not trivial, especially if the goal is that it be materially simpler than the protocol being offloaded to the remote processor. Historically, the HFEP has been the Achilles' heel of NFE processors. The HFEP ends up being asymptotically as complex as the "primary" protocol being offloaded, so there is very little to gain in offloading it. In addition, the HFEP must be implemented twice: once in the host and once in the front-end processor, each one of those being a different host platform as far as the HFEP is concerned. Two implementations, two integrations with host operating systems—this means twice as many sources of subtle race conditions, deadlocks, buffer starvations, and other nasty bugs. This cost requires a huge payoff to cover it.

### **But Wait a Minute...**

About now some readers may be eager to throw a penalty flag for "unconvincing hand waving" because even in the base case, there is a protocol between the Ethernet interface and the host

computer device driver. "Doesn't that count?" you rightfully ask. Yes, indeed, it does.

There is a long history of peripheral chips being designed with absolutely dreadful interfaces. Such chips have been known to make device-driver writers contemplate slow, painful violence if they ever meet the chip designer in a dark alley. The very early Ethernet chips from one famous semiconductor company were absolute masterpieces of egregious overdesign. Not only did they contain many complex functions of dubious utility, but also the functions that were genuinely required suffered from the same virulent infestation of bugs that plagued the useless bits. Tom Lyon wrote a famous Usenix paper in 1985, "All the Chips that Fit," delivering an epic rant on this expansive topic. (It should be required reading for anyone contemplating hardware design.)

If the goal is efficiency and performance of network code, all of the "mini-protocols" in the entire network protocol subsystem must be examined carefully. Both internal complexity and integration complexity can be serious bottlenecks. Ultimately, the question is how hard is it to glue this piece onto the other pieces it must interact with frequently? If it is very difficult, it is likely not fast (in an absolute sense), nor is it likely robust from a bug standpoint.

Remember the protocol state machines are generally not the principal source of complexity or performance issues. One extra data copy can make a huge difference in the maximum achievable performance. Therefore, implementations must focus on avoiding data motion: put it where it goes the first time it is touched, then leave it alone. If some other operation on packet payload is required, such as checksum computation, bury it inside an unavoidable operation such as the single transfer into memory. In line with those suggestions, streamline the operating-system interface to maximize concurrency. Once all those issues have been addressed aggressively, there's not a lot of work left to avoid.

### **What Does All this Mean for NFEs?**

Many times, but not every time, an NFE is likely to be an overly complex solution to the wrong part of the problem. It is possibly an expedient short-term

measure (and there's certainly a place in the world for those), but as a long-term architectural approach, the commoditization of processor cores makes specialized hardware very difficult to justify.

Lacking NFEs, what is required for maximizing host-based network performance? Here are some guidelines:

- Wire interfaces should be designed to be fast and brilliantly simple. Do the bit-speed work and then get the data into memory as quickly as possible, doing any additional work such as checksums that can readily be buried in the unavoidable transfer. Streamline the device as seen by the driver so as to avoid playing "Twenty Questions" with the hardware to determine what just happened.

- Interconnects should have sufficient capacity to carry the network traffic without strangling other I/O operations. From the standpoint of a network interface, PCI Express appears to have adequate performance for 10Gbps Ethernet as does HyperTransport 3.0.

- The system must have sufficient memory bandwidth to get the network payload in and out without strangling the rest of the system, especially the processors. Historically, the PC platform has been chronically starved for memory bandwidth.

- Processors should have enough cores able to exploit the sufficient memory bandwidth.

- Network protocol stacks should be designed to maximize parallelism and minimize blocking, while never copying data.

- A set of network APIs should be designed to maximize performance as opposed to mandatory similarity with existing system calls. Backward compatibility is important to support, but some applications may wish to pay more to get more.

### **Historical Perspective**

NFEs have been rediscovered in at least four or five different periods. In the spirit of full and fair disclosure, I must admit to having directly contributed to two of those efforts and having purchased and integrated yet another. So why does this idea keep recurring if it turns out to be much more difficult than it first appears?

The capacities and economics of computer systems do not advance smoothly, nor are the rates of improvement of various components synchronized. The resulting interactions produce dramatically different trade-offs in system partitioning that evolve over time. What is correct today may not be right after the next technology improvement. An example will illustrate the point.

Once upon a time, disk storage was expensive—really expensive—but it also exhibited significant economy of scale. At that time, LAN connectivity and processor performance were sufficient to make it desirable to share large disks among multiple workstations, giving rise to the *diskless workstation*. This lasted for a number of years, but as disks slid down the learning curve, the decreasing cost per megabyte of disk space overwhelmed the operational complexity of diskless workstations so they became diskfull, and they have been ever since—until relatively recently. Today the typical large organization averages the better part of one PC per employee, so the operational grief of administering all those desktop PCs is substantial. This cost is now high enough that the diskless workstation has been rediscovered, this time named *thin clients*. All the storage is elsewhere; nothing permanent exists on the desktop unit. History is busily repeating itself. Why? Because the various cost curves have moved enough, relative to each other, to the point where centralization makes sense.

The same thing happens with NFEs. At a point in time, systems don't have enough network "go-fast" to deliver the performance required, so just add a dedicated processor to the network interface to make up for it. The economics of that are fleeting at best, however. Between chip design and system-integration complexity, an NFE will need to be an economically attractive solution for quite some time to recoup the development costs. Unfortunately, the relentless improvements in processor, memory system, and system interconnect in the base PC platform make that window of advantage a shrinking, fast-moving target. Does anyone else remember the HiFN file compression processor chip? It was built into PC systems for a very short time. Proces-

sors quickly improved enough to do compression/decompression on the fly, however, and that was the end of HiFN's dream—well before the dropping cost of disk storage would have killed it.

Any effort to question the efficacy of NFEs should include a caveat for one particular case that merits a special mention because it indeed makes a compelling case for a particular style of NFE.

The proliferation of microcontrollers in devices such as thermostats, light switches, toasters, and almost everything else with more than a simple on/off switch has created a real opportunity for NFEs. Almost all of these microcontroller applications are typified by intense cost pressure, which usually translates into extreme limitations on available computing resources. It is simply out of the question to put a network stack in the vast majority of these systems, but the desirability of remote management of these devices increases daily.

This has created a new breed of NFE: the network communications adapter (NCA) that specializes in the simplicity of the protocol between the microcontroller host and the NFE—serial ASCII. Most microcontrollers have some serial port ability, so by looking like a terminal, the NCA can play the role of translator, speaking serial out one side and TCP/IP out the other. The NCA appears as a host on the TCP network, often containing a simple Web server that vends state information and may provide certain other management functions that get translated into simple ASCII exchanges with the microcontroller system.

An NCA is usually implemented in one of the more powerful microcontrollers that have been designed to provide an Ethernet interface and support enough RAM and ROM to contain a simplified network stack. The NCA is now available as an off-the-shelf module designed for easy integration no more difficult than a modem on a serial port.

The question of which is the tail and which is the dog comes to mind in many of these applications. From the TCP network's point of view, the NCA is the host and the microcontroller is being managed. From the point of view of

the lighting controller, the NCA looks like just one more switch, albeit a chaty one. This distinction is usually irrelevant—it just makes hash of pedantic layering diagrams. There's something quite satisfying about that.

## Conclusion

Rather than debate the religious propriety of NFEs, particularly the TOE variety, I have examined the architectural issues that have produced their recurring rise and fall. The TOE-style NFE is best viewed as a tactical tool with a limited expected lifetime of economic viability, not an enduring architectural approach. This is just another example of the recurring ebb and flow of functions between specialized peripherals and the system CPU(s), as the economics slosh back and forth interacting with system requirements. The limited lifetime of the NFE's advantages makes it difficult to justify the significant development costs for any but the highest-value applications.

That said, the inexpensive NCA is likely to be an approach that does endure. It literally transforms network communication into an inexpensive, pluggable physical component. By doing so, it provides an avenue for dealing with the extreme cost pressure inherent in microcontroller applications while providing an incremental option of genuine network citizenship when the customer will pay for it. □

### Related articles on queue.acm.org

#### TCP Offload to the Rescue

Andy Currid

<http://queue.acm.org/detail.cfm?id=1005069>

#### Network Virtualization

Scott Rixner

<http://queue.acm.org/detail.cfm?id=1348592>

#### DAFS: A New High-Performance Networked File System

Steve Kleiman

<http://queue.acm.org/detail.cfm?id=1388770>

**Mike O'Dell** is a venture partner at New Enterprise Associates (NEA), Chevy Chase, MD, where he works to identify early-stage IT, communications, and energy opportunities. Prior to this position, Odell was chief scientist at UUNET Technologies, responsible for network and product architecture during the emergence of the commercial Internet. He has also held positions at Bellcore (now Telcordia), a GaAs Sparc supercomputer startup, and a U.S. government contractor. He was founding editor of *Computing Systems*, an international refereed scholarly journal.



## High bandwidth, low latency, and multihoming challenge the sockets API.

BY GEORGE V. NEVILLE-NEIL

# Whither Sockets?

ONE OF THE most pervasive and longest-lasting interfaces in software is the sockets API. Developed by the Computer Systems Research Group at the University of California at Berkeley, the sockets API was first released as part of the 4.1c BSD operating system in 1982. While there are longer-lived APIs—

for example, those dealing with Unix file I/O—it is quite impressive for an API to have remained in use and largely unchanged for 27 years. The only major update to the sockets API has been the extension of ancillary routines to accommodate the larger addresses used by IPv6.<sup>2</sup>

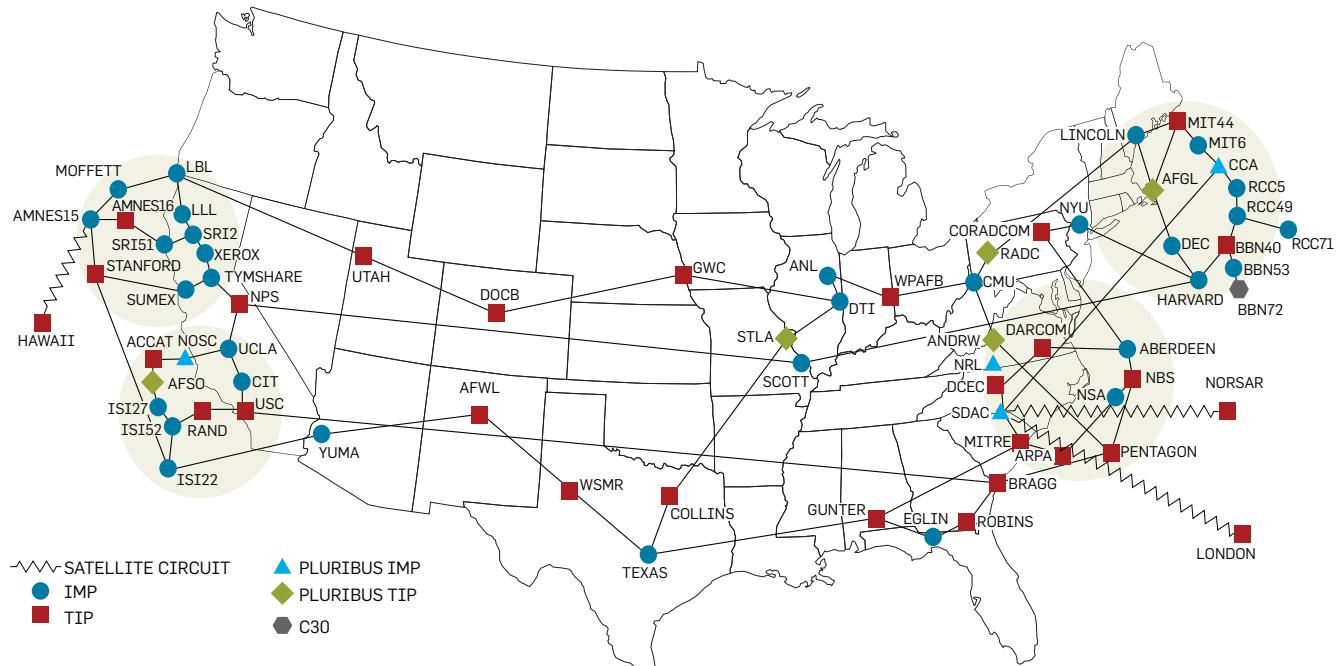
The Internet and the networking world in general have changed in very significant ways since the sockets API was first developed, but in many ways the API has had the effect of narrowing the way in which developers think about and write networked applications. This article briefly examines some of the conditions present when the sockets API was developed and considers how those conditions shaped the way in which networking code was written. Later, I look at ways in which developers have tried to get around some of the inherent limitations in the API and address the future of sockets in a changing networked world.

The two biggest differences between the networks of 1982 and 2009

are topology and speed. For the most part it is the increase in speed rather than the changes in topology that people notice. The maximum bandwidth of a commercially available long-haul network link in 1982 was 1.5Mbps. The Ethernet LAN, which was being deployed at the same time, had a speed of 10Mbps. A home user—and there were very few of these—was lucky to have a 300bps connection over a phone line to any computing facility. The round-trip time between two machines on a local area network was measured in tens of milliseconds, and between systems over the Internet in hundreds of milliseconds, depending of course on location and the number of hops a packet would be subjected to when being routed between machines. (See page 52 for a look at the early Internet.)

The topology of networks at the time was relatively simple. Most computers had a single connection to a local area network; the LAN was connected to a primitive router that might have a few connections to other LANs and a single

## ARPANET GEOGRAPHIC MAP, OCTOBER 1980



(Note: This map does not show ARPA's experimental satellite connections.)

Names shown are IMP names, not (necessarily) host names.

Source: <http://personalpages.manchester.ac.uk/staff/m.dodge/cybergeography/atlas/arpanet4.gif>.

connection to the Internet. For one application to another application, the connection was either across a LAN or transiting one or more routers, called IMPs (Internet message passing).

### History of Sockets

The model of distributed programming that came to be most popularized by the sockets API was the client/server model, in which there is a server and a set of clients. The clients send messages to the server to ask it to do work on their behalf, wait for the server to do the work requested, and at some later point receive an answer. This model of computing is now so ubiquitous it is often the only model with which many software engineers are familiar. At the time it was designed, however, it was

seen as a way of extending the Unix file I/O model over a computer network. One other factor that focused the sockets API down to the client/server model was that the most popular protocol it supported was TCP, which has an inherently 1:1 communication model.

The sockets API made the client/server model easy to implement because of the small number of extra system calls that programmers would need to add to their non-networked code so it could take advantage of other computing resources. Although other models are possible, with the sockets API the client/server model is the one that has come to dominate networked computing.

Although the sockets API has more entry points than those shown in Table

1, it is those five shown that are central to the API and that differentiate it from regular file I/O. In reality the `socket()` call could have been dropped and replaced with a variant of `open()`, but this was not done at the time. The `socket()` and `open()` calls actually return the same thing to a program: a process-unique file descriptor that is used in all subsequent operations with the API. It is the simplicity of the API that has led to its ubiquity, but that ubiquity has held back the development of alternative or enhanced APIs that could help programmers develop other types of distributed programs.

Client/server computing had many advantages at the time it was developed. It allowed many users to share resources, such as large storage arrays and expensive printing facilities, while keeping these facilities within the control of the same departments that had once run mainframe computing facilities. With this sharing model, it was possible to increase the utilization of what, at the time, were expensive resources.

Three disparate areas of networking are not well served by the sockets API: low-latency or real-time applications; high-bandwidth applications;

**Table 1: Socket API systems calls.**

<code>socket()</code>	Create a communication endpoint
<code>bind()</code>	Bind the endpoint to some set of network-layer parameters
<code>listen()</code>	Set a limit on the number of outstanding work requests
<code>accept()</code>	Accept one or more work requests from a client
<code>connect()</code>	Contact a server to submit a work request

and multihomed systems—that is, those with multiple network interfaces. Many people confuse increasing network bandwidth with higher performance, but increasing bandwidth does not necessarily reduce latency. The challenge for the sockets API is giving the application faster access to network data.

The way in which any program using the sockets API sends and receives data is via calls to the operating system. All of these calls have one thing in common: the calling program must repeatedly ask for data to be delivered. In a world of client/server computing these constant requests make perfect sense, because the server cannot do anything without a request from the client. It makes little sense for a print server to call a client unless the client has something it wishes to print. What, however, if the service provided is music or video distribution? In a media distribution service there may be one or more sources of data and many listeners. For as long as the user is listening to or viewing the media, the most likely case is that the application will want whatever data has arrived. Specifically requesting new data is a waste of time and resources for the application. The sockets API does not provide the programmer a way in which to say, “Whenever there is data for me, call me to process it directly.”

Sockets programs are instead written from the viewpoint of a dearth of, rather than a wealth of, data. Network programs are so used to waiting on data that they use a separate system call, `socket()`, so that they can listen to multiple sources of data without blocking on a single request. The typical processing loop of a sockets-based program isn’t simply `read(), process(), read()`, but instead `select(), read(), process(), select()`. Although the addition of a single system call to a loop would not seem to add much of a burden, this is not the case. Each system call requires arguments to be marshaled and copied into the kernel, as well as causing the system to block the calling process and schedule another. If there were data available to the caller when it invoked `select()`, then all of the work that went into crossing the user/kernel boundary was wasted because a `read()` would have returned data immediately. The

## Sockets programs are written from the viewpoint of a dearth of, rather than a wealth of, data.

constant check/read/check is wasteful unless the time between successive requests is quite long.

Solving this problem requires inverting the communication model between an application and the operating system. Various attempts to provide an API that allows the kernel to call directly into a program have been proposed but none has gained wide acceptance—for a few reasons. The operating systems that existed at the time the sockets API was developed were, except in very esoteric circumstances, single threaded and executed on single-processor computers. If the kernel had been fitted with an up-call API, there would have been the problem of which context the call could have executed in. Having all other work on a system pause because the kernel was executing an up-call into an application would have been unacceptable, particularly in timesharing systems with tens to hundreds of users. The only place in which such software architecture did gain currency was in embedded systems and networked routers where there were no users and no virtual memory.

The issue of virtual memory compounds the problems of implementing a kernel up-call mechanism. The memory allocated to a user process is virtual memory, but the memory used by devices such as network interfaces is physical. Having the kernel map physical memory from a device into a user-space program breaks one of the fundamental protections provided by a virtual memory system.

### Attempts to Overcome Performance Issues

A couple of different mechanisms have been proposed and sometimes implemented on various operating systems to overcome the performance issues present in the sockets API. One such mechanism is *zero-copy sockets*. Anyone who has worked on a network stack knows that copying data is what kills the performance of networking protocols. Therefore, to improve the speed of networked applications that are more interested in high bandwidth than in low latency, the operating system is modified to remove as many data copies as possible. Traditionally, an operating system performs two copies for each packet received by the system.

**Table 2: APIs added by SCTP.**

API	Explanation
sctp_bindx()	Bind or unbind an SCTP socket to a list of addresses
sctp_connectx()	Connect an SCTP socket with multiple destination addresses
sctp_generic_recvmsg()	Receive data from a peer
sctp_generic_sendmsg(), sctp_generic_sendmsg iov()	Send data to a peer
sctp_getaddrlen()	Return the address length of an address family
sctp_getassocid()	Return an association ID for a specified socket address
sctp_getpaddrs(), sctp_getladdrs()	Return list of addresses to caller
sctp_peeloff()	Detach an association from a one-to-many socket to a separate file descriptor
sctp_sendx()	Send a message from an SCTP socket
sctp_sendmsgx()	Send a message from an SCTP socket

The first copy is performed by the network driver from the network device's memory into the kernel's memory, and the second is performed by the sockets layer in the kernel when the data is read by the user program. Each of these copy operations is expensive because it must occur for each message that the system receives. Similarly, when the program wants to send a message, data must be copied from the user's program into the kernel for each message sent; then that data will be copied into the buffers used by the device to transmit it on the network.

Most operating-system designers and developers know that data copying is anathema to system performance and work to minimize such copies *within the kernel*. The easiest way for the kernel to avoid a data copy is to have device drivers copy data directly into and out of kernel memory. On modern network devices this is a result of how they structure their memory. The driver and kernel share two rings of packet descriptors—one for transmit and one for receive—where each descriptor has a single pointer to memory. The network device driver initially fills these rings with memory from the kernel. When data is received, the device sets a flag in the correct receive descriptor and tells the kernel, usually via an interrupt, that there is data waiting for it. The kernel then removes the filled buffer from the receive descriptor ring and replaces it with a fresh buffer for the device to fill. The packet, in the form of the buffer, then moves through the network stack

until it reaches the socket layer, where it is *copied* out of the kernel when the user's program calls `read()`. Data sent by the program is handled in a similar way by the kernel, in that kernel buffers are eventually added to the transmit descriptor ring and a flag is then set to tell the device that it can place the data in the buffer on the network.

All of this work in the kernel leaves the last copy problem unsolved, and several attempts have been made to extend the sockets API to remove this copy operation.<sup>1, 3</sup> The problem remains as to how memory can be safely shared across the user/kernel boundary. The kernel cannot give its memory over to the user program, because at that point it loses control over the memory. A user program that crashes may leave the kernel without a significant chunk of usable memory, leading to system performance degradation. There are also security issues inherent in sharing memory buffers across the kernel/user boundary. There is no single answer to how a user program might achieve higher bandwidth using the sockets API.

For programmers who are more concerned with latency than with bandwidth, even less has been done. The only significant improvement for programs that are waiting for a network event has been the addition of a set of kernel events that a program can wait on. Kernel events, or `kevents()`, are an extension of the `select()` mechanism to encompass any possible event that the kernel might be able to tell the program about. Before the advent of `kevents`, a user program could call

`select()` on any file descriptor, which would let the program know when any of a set of file descriptors was readable, writable, or had an error. When programs were written to sit in a loop and wait on a set of file descriptors—for example, reading from the network and writing to disk—the `select()` call was sufficient, but once a program wanted to check for other events, such as timers and signals, `select()` no longer served. The problem for low-latency apps is that `kevents()` do not deliver data; they deliver only a signal that data is ready, just as the `select()` call did. The next logical step would be to have an event-based API that also delivered data. There is no reason to have the application cross the user/kernel boundary twice simply to get the data the kernel knows the application wants.

### Lack of Support for Multihoming

The sockets API not only presents performance problems to the application writer, but also narrows the type of communication that can take place. The client/server paradigm is inherently a 1:1 type of communication. Although a server may handle requests from a diverse group of clients, each client has only one connection to a single server for a request or set of requests. In a world in which each computer had only one network interface, that paradigm made perfect sense. A connection between a client and server is identified by a quad of <Source IP, Source Port, Destination IP, Destination Port>. Since services generally have a well-known destination port (for example, 80 for HTTP), the only value that can easily vary is the source port, since the IP addresses are fixed.

In the Internet of 1982 each machine that was not a router had only a single network interface, meaning that to identify a service, such as a remote printer, the client computer needed a single destination address and port and had, itself, only a single source address and port to work with. While it did exist, the idea that a computer might have multiple ways of reaching a service was too complicated and far too expensive to implement. Given these constraints, there was no reason for the sockets API to expose to the programmer the ability to write a multihomed program—one that could manage

which interfaces or connections mattered to it. Such features, when they were implemented, were a part of the routing software within the operating system. The only way programs could get access to them was through an obscure set of nonstandard kernel APIs called a routing socket.

On a system with multiple network interfaces it is not possible, using the standard sockets API, to write an application that can easily be multihomed—that is, take advantage of both interfaces so if one fails, or if the primary route over which the packets were flowing breaks, the application would not lose its connection to the server.

The recently developed Stream Control Transport Protocol (SCTP)<sup>4</sup> incorporates support for multihoming at the protocol level, but it is impossible to export this support through the sockets API. Several ad-hoc system calls were initially provided and are the only way to access this functionality. At the moment this is the only protocol that has both the capacity and user demand for this feature, so the API has not been standardized across more than a few operating systems. Table 2 shows the APIs that SCTP added.

While the list of functions in Table 2 contains more APIs than are strictly necessary, it is important to note that many are derivatives of preexisting APIs, such as `send()`, which need to be extended to work in a multihoming world. The set of APIs needs to be harmonized to make multihoming a first-class citizen in the sockets world. The problem now is that sockets are so successful and ubiquitous that it is very hard to change the existing API set for fear of confusing its users or the preexisting programs that use it.

As systems come to have more network interfaces built in, providing the ability to write applications that take advantage of multihoming will be an absolute necessity. One can easily imagine the use of such technology in a smartphone, which already has three network interfaces: its primary connection via the cellular network, a WiFi interface, and often a Bluetooth interface as well. There is no reason for an application to lose connectivity if even one of these network interfaces is working properly. The problem for application designers is that they want their code

## As systems come to have more network interfaces built in, providing the ability to write applications that take advantage of multihoming will be an absolute necessity.

to work, with few or no changes, across a plethora of devices, from cellphones, to laptops, to desktops, and so on. With properly defined APIs we would remove the artificial barrier that prevents this. It is only because of the history of the sockets API and the fact that it has been “good enough” to date that this need has not yet been addressed.

High bandwidth, low latency, and multihoming are driving the development of alternatives to the sockets API. With LANs now reaching 10Gbps, it is obvious that for many applications client/server style communication is far too inefficient to use the available bandwidth. The communication paradigms supported by the sockets API must be expanded to allow for memory sharing across the kernel boundary, as well as for lower-latency mechanisms to deliver data to applications. Multihoming must become a first-class feature of the sockets API because devices with multiple active interfaces are now becoming the norm for networked systems. □

### Related articles on queue.acm.org

#### **Code Spelunking: Exploring Cavernous Code Bases**

George Neville-Neil  
<http://queue.acm.org/detail.cfm?id=945136>

#### **API Design Matters**

Michi Henning  
<http://queue.acm.org/detail.cfm?id=1255422>

#### **You Don't Know Jack about Network Performance**

Kevin Fall and Steve McCanne  
<http://queue.acm.org/detail.cfm?id=1066069>

### References

1. Balaji, P., Bhagvat, S., Jin, H.-W., and Panda, D.K. Asynchronous zero-copy communication for synchronous sockets in the sockets direct protocol (sdp) over infiniband journal. In *Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium*.
2. Gilligan, R., Thomson, S., Bound, J., McCann, J., and Stevens, W. Basic Socket Interface Extensions for IPv6. RFC 3493 (Feb. 2003); <http://www.rfc-editor.org/rfc/rfc3493.txt>.
3. Romanow, A., Mogul, J., Talpey, T., and Bailey, S. Remote Direct Memory Access (RDMA) over IP Problem Statement. RFC 4297 (Dec. 2005); <http://www.rfc-editor.org/rfc/rfc4297.txt>.
4. Stewart, R., et al. Stream Control Transmission Protocol. RFC 2960 (Oct. 2000); <http://www.ietf.org/rfc/rfc2960.txt>.

**George V. Neville-Neil** ([kv@acm.org](mailto:kv@acm.org)) is the proprietor of Neville-Neil Consulting. He works on networking and operating systems code and teaches courses on program-related topics.

© 2009 ACM 0001-0782/09/0600 \$10.00

# contributed articles

DOI:10.1145/1516046.1516062

RAKESH AGRAWAL  
ANASTASIA AILAMAKI  
PHILIP A. BERNSTEIN  
ERIC A. BREWER  
MICHAEL J. CAREY  
SURAJIT CHAUDHURI  
ANHAI DOAN  
DANIELA FLORESCU  
MICHAEL J. FRANKLIN  
HECTOR GARCIA-MOLINA  
JOHANNES GEHRKE  
LE GRUENWALD  
LAURA M. HAAS  
ALON Y. HALEVY  
JOSEPH M. HELLERSTEIN  
YANNIS E. IOANNIDIS  
HANK F. KORTH  
DONALD KOSSMANN  
SAMUEL MADDEN  
ROGER MAGOULAS  
BENG CHIN OOI  
TIM O'REILLY  
RAGHU RAMAKRISHNAN  
SUNITA SARAWAGI  
MICHAEL STONEBRAKER  
ALEXANDER S. SZALAY  
GERHARD WEIKUM

**Database research is expanding, with major efforts in system architecture, new languages, cloud services, mobile and virtual worlds, and interplay between structure and text.**

## The Claremont Report on Database Research

A GROUP OF database researchers, architects, users, and pundits met in May 2008 at the Claremont Resort in Berkeley, CA, to discuss the state of database research and its effects on practice. This was the seventh meeting of this sort over the past 20 years and was distinguished by a broad consensus that the database community is at a turning point in its history, due to both an explosion of data and usage scenarios and major shifts in computing hardware and platforms.

Here, we explore the conclusions of this self-assessment. It is by definition somewhat inward-focused but may be of interest to the broader computing community as both a window into upcoming directions in database research and

a description of some of the community issues and initiatives that surfaced. We describe the group's consensus view of new focus areas for research, including database engine architectures, declarative programming languages, interplay of structured data and free text, cloud data services, and mobile and virtual worlds. We also report on discussions of the database community's growth and processes that may be of interest to other research areas facing similar challenges.

Over the past 20 years, small groups of database researchers have periodically gathered to assess the state of the field and propose directions for future research.<sup>1,3–7</sup> Reports of the meetings served to foster debate within the database research community, explain research directions to external organizations, and help focus community efforts on timely challenges.

The theme of the Claremont meeting was that database research and the data-management industry are at a turning point, with unusually rich opportunities for technical advances, intellectual achievement, entrepreneurship, and benefits for science and society. Given the large number of opportunities, it is important for the database research community to address issues that maximize relevance within the field, across computing, and in external fields as well.

The sense of change that emerged in the meeting was a function of several factors:

**Excitement over “big data.”** In recent years, the number of communities working with large volumes of data has grown considerably to include not only traditional enterprise applications and Web search but also e-science efforts (in astronomy, biology, earth science, and more), digital entertainment, natural-language processing, and social-network analysis. While the user base for traditional database management systems (DBMSs) is growing quickly, there is also a groundswell of effort to design new custom data-management solutions from simpler components. The ubiquity of big data is expanding the base of users and developers of data-management technologies and will undoubtedly shake up the database research field.

*Data analysis as profit center.* In tradi-

tional enterprise settings, the barriers between IT departments and business units are coming down, and there are many examples of companies where data is indeed the business itself. As a consequence, data capture, integration, and analysis are no longer viewed as a business cost but as the keys to efficiency and profit. The value of software to support data analytics has been growing as a result. In 2007, corporate acquisitions of business-intelligence vendors alone totaled \$15 billion,<sup>2</sup> and

crawls of deep-Web sites. There is also an explosion of text-focused semistructured data in the public domain in the form of blogs, Web 2.0 communities, and instant messaging. New incentive structures and Web sites have emerged for publishing and curating structured data in a shared fashion as well. Text-centric approaches to managing the data are easy to use but ignore latent structure in the data that might add significant value. The race is on to develop techniques that extract useful



that is only the “front end” of the data-analytics tool chain. Market pressure for better analytics also brings new users to the technology with new demands. Statistically sophisticated analysts are being hired in a growing number of industries, with increasing interest in running their formulae on the raw data. At the same time, a growing number of nontechnical decision makers want to “get their hands on the numbers” as well in simple and intuitive ways.

**Ubiquity of structured and unstructured data.** There is an explosion of structured data on the Web and on enterprise intranets. This data is from a variety of sources beyond traditional databases, including large-scale efforts to extract structured information from text, software logs and sensors, and

data from mostly noisy text and structured corpora, enable deeper exploration into individual data sets, and connect data sets together to wring out as much value as possible.

**Expanded developer demands.** Programmer adoption of relational DBMSs and query languages has grown significantly in recent years, accelerated by the maturation of open source systems (such as MySQL and PostgreSQL) and the growing popularity of object-relational mapping packages (such as Ruby on Rails). However, the expanded user base brings new expectations for programmability and usability from a larger, broader, less-specialized community of programmers.

Some of them are unhappy or unwilling to “drop into” SQL, viewing DBMSs

as unnecessarily complicated and daunting to learn and manage relative to other open source components. As the ecosystem for database management evolves beyond the typical DBMS user base, opportunities are emerging for new programming models and new system components for data management and manipulation.

*Architectural shifts in computing.* While the variety of user scenarios is increasing, the computing substrates for data management are shifting

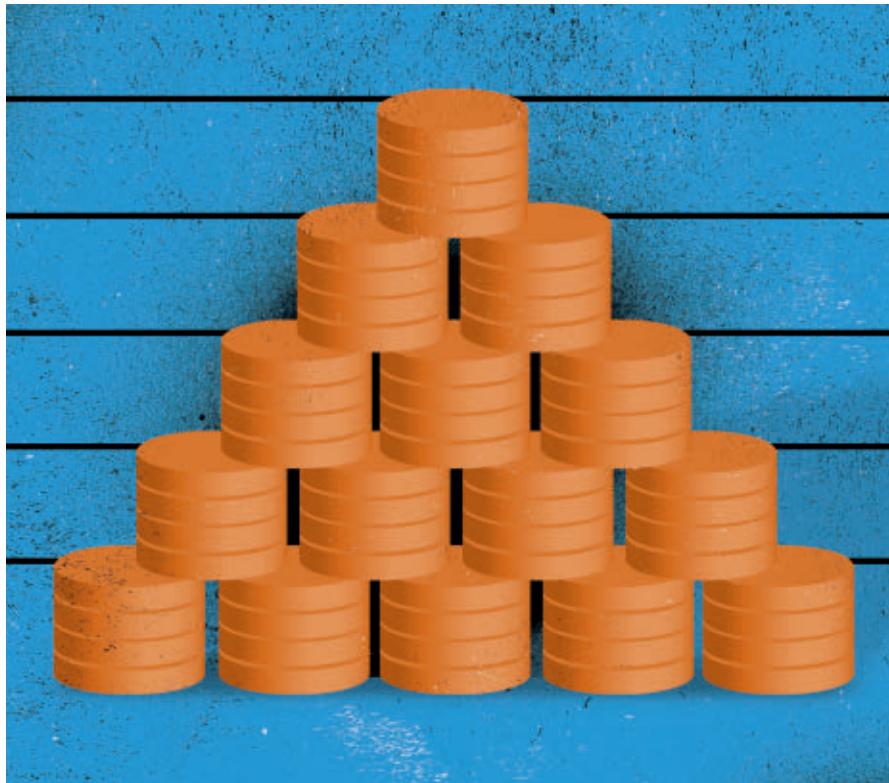
tant aspect of the price/performance metric of large systems. These hardware trends alone motivate a wholesale reconsideration of data-management software architecture.

These factors together signal an urgent, widespread need for new data-management technologies. There is an opportunity for making a positive difference. Traditionally, the database community is known for the practical relevance of its research; relational databases are emblematic of technol-

revolved around two broad agendas we call reformation and synthesis. The reformation agenda involves deconstructing traditional data-centric ideas and systems and reforming them for new applications and architectural realities. One part of this entails focusing outside the traditional RDBMS stack and its existing interfaces, emphasizing new data-management systems for growth areas (such as e-science). Another part of the reformation agenda involves taking data-centric ideas like declarative programming and query optimization outside their original context in storage and retrieval to attack new areas of computing where a data-centric mindset promises to yield significant benefit. The synthesis agenda is intended to leverage research ideas in areas that have yet to develop identifiable, agreed-upon system architectures, including data integration, information extraction, and data privacy. Many of these subcommunities of database research seem ready to move out of the conceptual and algorithmic phase to work together on comprehensive artifacts (such as systems, languages, and services) that combine multiple techniques to solve complex user problems. Efforts toward synthesis can serve as rallying points for research, likely leading to new challenges and breakthroughs, and promise to increase the overall visibility of the work.

## Research Opportunities

After two days of intense discussion at the 2008 Claremont meeting, it was surprisingly easy for the group to reach consensus on a set of research topics for investigation in coming years. Before exploring them, we stress a few points regarding what is not on the list. First, while we tried to focus on new opportunities, we do not propose they be pursued at the expense of existing good work. Several areas we deemed critical were left off because they are already focus topics in the database community. Many were mentioned in previous reports<sup>1,3–7</sup> and are the subject of significant efforts that require continued investigation and funding. Second, we kept the list short, favoring focus over coverage. Though most of us have other promising research topics we would have liked to discuss at greater length here, we focus on topics that



dramatically as well. At the macro scale, the rise of cloud computing services suggests fundamental changes in software architecture. It democratizes access to parallel clusters of computers; every programmer has the opportunity and motivation to design systems and services that scale out incrementally to arbitrary degrees of parallelism. At a micro scale, computer architectures have shifted the focus of Moore's Law from increasing clock speed per chip to increasing the number of processor cores and threads per chip. In storage technologies, major changes are under way in the memory hierarchy due to the availability of more and larger on-chip caches, large inexpensive RAM, and flash memory. Power consumption has become an increasingly impor-

ty transfer. But in recent years, the externally visible contribution of the database research community has not been as pronounced, and there is a mismatch between the notable expansion of the community's portfolio and its contribution to other fields of research and practice. In today's increasingly rich technical climate, the database community must recommit itself to impact and breadth. Impact is evaluated by external measures, so success involves helping new classes of users, powering new computing platforms, and making conceptual breakthroughs across computing. These should be the motivating goals for the next round of database research.

To achieve these goals, discussion at the 2008 Claremont Resort meeting

attracted the broadest interest within the group.

In addition to the listed topics, the main issues raised during the meeting included management of uncertain information, data privacy and security, e-science and other scholarly applications, human-centric interaction with data, social networks and Web 2.0, personalization and contextualization of query- and search-related tasks, streaming and networked data, self-tuning and adaptive systems, and the challenges raised by new hardware technologies and energy constraints. Most are captured in the following discussion, with many cutting across multiple topics.

*Revisiting database engines.* System R and Ingres pioneered the architecture and algorithms of relational databases; current commercial databases are still based on their designs. But many of the changes in applications and technology demand a reformation of the entire system stack for data management. Current big-market relational database systems have well-known limitations. While they provide a range of features, they have only narrow regimes in which they provide peak performance; online transaction processing (OLTP) systems are tuned for lots of small, concurrent transactional debit/credit workloads, while decision-support systems are tuned for a few read-mostly, large-join-and-aggregation workloads. Meanwhile, for many popular data-intensive tasks developed over the past decade, relational databases provide poor price/performance and have been rejected; critical scenarios include text indexing, serving Web pages, and media delivery. New workloads are emerging in the sciences, Web 2.0-style applications, and other environments where database-engine technology could prove useful but is not bundled in current database systems.

Even within traditional application domains, the database marketplace today suggests there is room for significant innovation. For example, in the analytics markets for business and science, customers can buy petabytes of storage and thousands of processors, but the dominant commercial database systems typically cannot scale that far for many workloads. Even when they can, the cost of software and

## The ubiquity of big data is expanding the base of users and developers of data-management technologies and will undoubtedly shake up the database research field.

management relative to hardware is exorbitant. In the OLTP market, business imperatives like regulatory compliance and rapid response to changing business conditions raise the need to address data life-cycle issues (such as data provenance, schema evolution, and versioning).

Given these requirements, the commercial database market is wide open to new ideas and systems, as reflected in the recent funding climate for entrepreneurs. It is difficult to recall when there were so many start-up companies developing database engines, and the challenging economy has not trimmed the field much. The market will undoubtedly consolidate over time, but things are changing fast, and it remains a good time to try radical ideas.

Some research projects have begun taking revolutionary steps in database system architecture. There are two distinct directions: broadening the useful range of applicability for multipurpose database systems (for example, to incorporate streams, text search, XML, and information integration) and radically improving performance by designing special-purpose database systems for specific domains (for example, read-mostly analytics, streams, and XML). Both directions have merit, and the overlap in their stated targets suggests they may be more synergistic than not. Special-purpose techniques (such as new storage and compression formats) may be reusable in more general-purpose systems, and general-purpose architectural components (such as extensible query optimizer frameworks) may help speed prototyping of new special-purpose systems.

Important research topics in the core database engine area include:

- ▶ Designing systems for clusters of many-core processors that exhibit limited and nonuniform access to off-chip memory;
- ▶ Exploiting remote RAM and Flash as persistent media, rather than relying solely on magnetic disk;
- ▶ Treating query optimization and physical data layout as a unified, adaptive, self-tuning task to be carried out continuously;
- ▶ Compressing and encrypting data at the storage layer, integrated with data layout and query optimization;

- ▶ Designing systems that embrace nonrelational data models, rather than shoehorning them into tables;
- ▶ Trading off consistency and availability for better performance and thousands of machines; and
- ▶ Designing power-aware DBMSs that limit energy costs without sacrificing scalability.

This list is not exhaustive. One industrial participant at the Claremont meeting noted that this is a time of opportunity for academic researchers; the landscape has shifted enough that access to industrial legacy code provides little advantage, and large-scale clustered hardware is rentable in the cloud at low cost. Moreover, industrial players and investors are aggressively looking for bold new ideas. This opportunity for academics to lead in system design is a major change in the research environment.

*Declarative programming for emerging platforms.* Programmer productivity is a key long-acknowledged challenge in computing, with its most notable mention in the database context in Jim Gray's 1998 Turing lecture. Today, the urgency of the challenge is increasing exponentially as programmers target ever more complex environments, including many-core chips, distributed services, and cloud computing platforms.

Nonexpert programmers must be able to write robust code that scales out across processors in both loosely and tightly coupled architectures. Although developing new programming paradigms is not a database problem per se, ideas of data independence, declarative programming, and cost-based optimization provide a promising angle of attack. There is significant evidence that data-centric approaches will have significant influence on programming in the near term.

The recent popularity of the MapReduce programming framework for manipulating big data sets is an example of this potential. MapReduce is attractively simple, building on language and data-parallelism techniques that have been known for decades. For database researchers, the significance of MapReduce is in demonstrating the benefits of data-parallel programming to new classes of developers.

**This is a unique opportunity for a fundamental “reformation” of the notion of data management, not as a single system but as a set of services that can be embedded, as needed, in many computing contexts.**

This opens opportunities for the database community to extend its contribution to the broader community, developing more powerful and efficient languages and runtime mechanisms that help these developers address more complex problems.

As another example of declarative programming, in the past five years a variety of new declarative languages, often grounded in Datalog, have been developed for domain-specific systems in fields as diverse as networking and distributed systems, computer games, machine learning and robotics, compilers, security protocols, and information extraction. In many of these scenarios, the use of a declarative language has reduced code size by orders of magnitude while also enabling distributed or parallel execution. Surprisingly, the groups behind these efforts have coordinated very little with one another; the move to revive declarative languages in these new contexts has grown up organically.

A third example arises in enterprise-application programming. Recent language extensions (such as Ruby on Rails and LINQ) encourage query-like logic in programmer design patterns. But these packages have yet to address the challenge of enterprise-style programming across multiple machines; the closest effort here is DryadLINQ, focusing on parallel analytics rather than on distributed application development. For enterprise applications, a key distributed design decision is the partitioning of logic and data across multiple “tiers,” including Web clients, Web servers, application servers, and a backend DBMS. Data independence is particularly valuable here, allowing programs to be specified without making a priori permanent decisions about physical deployment across tiers. Automatic optimization processes could make these decisions and move data and code as needed to achieve efficiency and correctness. XQuery has been proposed as an existing language that would facilitate this kind of declarative programming, in part because XML is often used in cross-tier protocols.

It is unusual to see this much energy surrounding new data-centric programming techniques, but the opportunity brings challenges as

well. The research challenges include language design, efficient compilers and runtimes, and techniques to optimize code automatically across both the horizontal distribution of parallel processors and the vertical distribution of tiers. It seems natural that the techniques behind parallel and distributed databases—partitioned dataflow and cost-based query optimization—should extend to new environments. However, to succeed, these languages must be fairly expressive, going beyond simple MapReduce and select-project-join-aggregate dataflows. This agenda will require “synthesis” work to harvest useful techniques from the literature on database and logic programming languages and optimization, as well as to realize and extend them in new programming environments.

To genuinely improve programmer productivity, these new approaches also need to pay attention to the softer issues that capture the hearts and minds of programmers (such as attractive syntax, typing and modularity, development tools, and smooth interaction with the rest of the computing ecosystem, including networks, files, user interfaces, Web services, and other languages). This work also needs to consider the perspective of programmers who want to use their favorite programming languages and data services as primitives in those languages. Example code and practical tutorials are also critical.

To execute successfully, database research must look beyond its traditional boundaries and find allies throughout computing. This is a unique opportunity for a fundamental “reformation” of the notion of data management, not as a single system but as a set of services that can be embedded as needed in many computing contexts.

*Interplay of structured and unstructured data.* A growing number of data-management scenarios involve both structured and unstructured data. Within enterprises, we see large heterogeneous collections of structured data linked with unstructured data (such as document and email repositories). On the Web, we also see a growing amount of structured data primarily from three sources: millions of databases hidden behind forms (the deep Web); hundreds of millions of high-

quality data items in HTML tables on Web pages and a growing number of mashups providing dynamic views on structured data; and data contributed by Web 2.0 services (such as photo and video sites, collaborative annotation services, and online structured-data repositories).

A significant long-term goal for the database community is to transition from managing traditional databases consisting of well-defined schemata for structured business data to the

it developed domain-independent technology for crawling through forms (that is, automatically submitting well-formed queries to forms) and surfacing the resulting HTML pages in a search-engine index. Within the enterprise, the database research community recently contributed to enterprise search and the discovery of relationships between structured and unstructured data.

The first challenge database researchers face is how to extract struc-



much more challenging task of managing a rich collection of structured, semi-structured, and unstructured data spread over many repositories in the enterprise and on the Web—sometimes referred to as the challenge of managing dataspaces.

In principle, this challenge is closely related to the general problem of data integration, a longstanding area for database research. The recent advances in this area and the new issues due to Web 2.0 resulted in significant discussion at the Claremont meeting. On the Web, the database community has contributed primarily in two ways: First, it developed technology that enables the generation of domain-specific (“vertical”) search engines with relatively little effort; and second,

ture and meaning from unstructured and semistructured data. Information-extraction technology can now pull structured entities and relationships out of unstructured text, even in unsupervised Web-scale contexts. We expect in coming years that hundreds of extractors will be applied to a given data source. Hence developers and analysts need techniques for applying and managing predictions from large numbers of independently developed extractors. They also need algorithms that can introspect about the correctness of extractions and therefore combine multiple pieces of extraction evidence in a principled fashion. The database community is not alone in these efforts; to contribute in this area, database researchers should continue

to strengthen ties with researchers in information retrieval and machine learning.

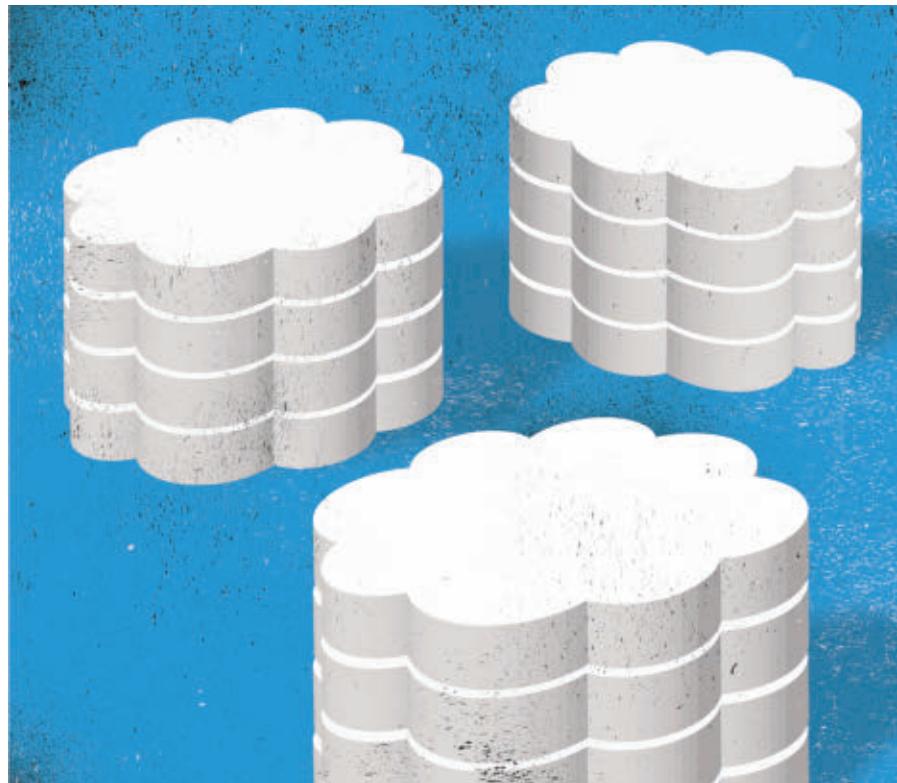
Context is a significant aspect of the semantics of the data, taking multiple forms (such as the text and hyperlinks that surround a table on a Web page, the name of the directory in which data is stored, accompanying annotations or discussions, and relationships to physically or temporally proximate data items). Context helps analysts interpret the meaning

develop methods to answer keyword queries over large collections of heterogeneous data sources. We must be able to break down the query to extract its intended semantics and route the query to the relevant source(s) in the collection. Keyword queries are just one entry point into data exploration, and there is a need for techniques that lead users into the most appropriate querying mechanism. Unlike previous work on information integration, the challenges here are that we cannot

concepts around which these functionalities are tied.

In addition to managing existing data collections, there is an opportunity to innovate in the creation of data collections. The emergence of Web 2.0 creates the potential for new kinds of data-management scenarios in which users join ad hoc communities to create, collaborate, curate, and discuss data online. As an example, consider creating a database of access to clean water in different places around the world. Since such communities rarely agree on schemata ahead of time, the schemata must be inferred from the data; however, the resulting schemata are still used to guide users to consensus. Systems in this context must incorporate visualizations that drive exploration and analysis. Most important, these systems must be extremely easy to use and so will probably require compromising on some typical database functionality and providing more semiautomatic "hints" mined from the data. There is an important opportunity for a feedback loop here; as more data is created with such tools, information extraction and querying could become easier. Commercial and academic prototypes are beginning to appear, but there is plenty of room for additional innovation and contributions.

*Cloud data services.* Economic and technological factors have motivated a resurgence of shared computing infrastructure, providing software and computing facilities as a service, an approach known as cloud services or cloud computing. Cloud services provide efficiencies for application providers by limiting up-front capital expenses and by reducing the cost of ownership over time. Such services are typically hosted in a data center using shared commodity hardware for computation and storage. A varied set of cloud services is available today, including application services ([salesforce.com](http://salesforce.com)), storage services ([Amazon S3](http://Amazon S3)), compute services ([Amazon EC2](http://Amazon EC2), Google App Engine, and Microsoft Azure), and data services ([Amazon SimpleDB](http://Amazon SimpleDB), Microsoft SQL Data Services, and Google's Datastore). They represent a major reformation of data-management architectures, with more on the horizon. We anticipate many future data-centric applications lever-



of data in such applications because the data is often less precise than in traditional database applications, as it is extracted from unstructured text, extremely heterogeneous, or sensitive to the conditions under which it was captured. Better database technology is needed to manage data in context. In particular, there is a need for techniques to discover data sources, enhance the data by discovering implicit relationships, determine the weight of an object's context when assigning it semantics, and maintain the provenance of data through these steps of storage and computation.

The second challenge is to develop methods for querying and deriving insight from the resulting sea of heterogeneous data. A specific problem is to

assume we have semantic mappings for the data sources and we cannot assume that the domain of the query or the data sources is known. We need to develop algorithms for providing best-effort services on loosely integrated data. The system should provide meaningful answers to queries with no need for manual integration and improve over time in a pay-as-you-go fashion as semantic relationships are discovered and refined. Developing index structures to support querying hybrid data is also a significant challenge. More generally, we need to develop new notions of correctness and consistency in order to provide metrics and enable users or system designers to make cost/quality trade-offs. We also need to develop the appropriate systems

aging data services in the cloud.

A cross-cutting theme in cloud services is the trade-off providers face between functionality and operational costs. Today's early cloud data services offer an API that is much more restricted than that of traditional database systems, with a minimalist query language, limited consistency guarantees, and in some cases explicit constraints on resource utilization. This limited functionality pushes more programming burden on developers but allows cloud providers to build more predictable services and offer service-level agreements that would be difficult to provide for a full-function SQL data service. More work and experience are needed on several fronts to fully understand the continuum between today's early cloud data services and more full-function but possibly less-predictable alternatives.

Manageability is particularly important in cloud environments. Relative to traditional systems, it is complicated by three factors: limited human intervention, high-variance workloads, and a variety of shared infrastructures. In the majority of cloud-computing settings, there will be no database administrators or system administrators to assist developers with their cloud-based applications; the platform must do much of that work automatically. Mixed workloads have always been difficult to tune but may be unavoidable in this context.

Even a single customer's workload can vary widely over time; the elastic provisioning of cloud services makes it economical for a user to occasionally harness orders-of-magnitude more resources than usual for short bursts of work. Meanwhile, service tuning depends heavily on the way the shared infrastructure is "virtualized." For example, Amazon EC2 uses hardware-level virtual machines as its programming interface. On the opposite end of the spectrum, salesforce.com implements "multi-tenant" hosting of many independent schemas in a single managed DBMS. Many other virtualization solutions are possible, each with different views into the workloads above and platforms below and different abilities to control each. These variations require revisiting traditional roles and responsibilities for resource

## Limited functionality pushes more programming burden on developers but allows cloud providers to build more predictable services and offer service-level agreements that would be difficult to provide for a full-function SQL data service.

management across layers.

The need for manageability adds urgency to the development of self-managing database technologies that have been explored over the past decade. Adaptive, online techniques will be required to make these systems viable, while new architectures and APIs, including the flexibility to depart from traditional SQL and transactional semantics when prudent, reduce requirements for backward compatibility and increase the motivation for aggressive redesign.

The sheer scale of cloud computing involves its own challenges. Today's SQL databases were designed in an era of relatively reliable hardware and intensive human administration; as a result, they do not scale effectively to thousands of nodes being deployed in a massively shared infrastructure. On the storage front, it is unclear whether these limitations should be addressed with different transactional implementation techniques, different storage semantics, or both simultaneously. The database literature is rich in proposals on these issues. Cloud services have begun to explore simple pragmatic approaches, but more work is needed to synthesize ideas from the literature in modern cloud computing regimes. In terms of query processing and optimization, it will not be feasible to exhaustively search a domain that considers thousands of processing sites, so some limitations on either the domain or the search will be required. Finally, it is unclear how programmers will express their programs in the cloud, as discussed earlier.

The sharing of physical resources in a cloud infrastructure puts a premium on data security and privacy that cannot be guaranteed by physical boundaries of machines or networks. Hence cloud services are fertile ground for efforts to synthesize and accelerate the work the database community has done in these areas. The key to success is to specifically target usage scenarios in the cloud, seated in practical economic incentives for service providers and customers.

As cloud data services become popular, new scenarios will emerge with their own challenges. For example, we anticipate specialized services that are pre-loaded with large data sets (such as

stock prices, weather history, and Web crawls). The ability to “mash up” interesting data from private and public domains will be increasingly attractive and provide further motivation for the challenges discussed earlier concerning the interplay of structured and unstructured data. The desire to mash up data also points to the inevitability of services reaching out across clouds, an issue already prevalent in scientific data “grids” that typically have large shared data servers at multiple sites, even within a single discipline. It also echoes, in the large, the standard proliferation of data sources in most enterprises. Federated cloud architectures will only add to these challenges.

*Mobile applications and virtual worlds.* This new class of applications, exemplified by mobile services and virtual worlds, is characterized by the need to manage massive amounts of diverse user-created data, synthesize it intelligently, and provide real-time services. The database community is beginning to understand the challenges faced by these applications, but much more work is needed. Accordingly, the discussion about these topics at the meeting was more speculative than about those of the earlier topics but still deserve attention.

Two important trends are changing the nature of the field. First, the platforms on which mobile applications are built—hardware, software, and network—have attracted large user bases and ubiquitously support powerful interactions “on the go.” Second, mobile search and social networks suggest an exciting new set of mobile applications that can deliver timely information (and advertisements) to mobile users depending on location, personal preferences, social circles, and extraneous factors (such as weather), as well as the context in which they operate. Providing these services requires synthesizing user input and behavior from multiple sources to determine user location and intent.

The popularity of virtual worlds like Second Life has grown quickly and in many ways mirrors the themes of mobile applications. While they began as interactive simulations for multiple users, they increasingly blur the distinctions with the real world and suggest the potential for a more

## Electronic media underscore the modern reality that it is easy to be widely published but much more difficult to be widely read.

data-rich mix. The term “co-space” is sometimes used to refer to a coexisting space for both virtual and physical worlds. In it, locations and events in the physical world are captured by a large number of sensors and mobile devices and materialized within a virtual world. Correspondingly, certain actions or events within the virtual world affect the physical world (such as shopping, product promotion, and experiential computer gaming). Applications of co-space include rich social networking, massive multi-player games, military training, edutainment, and knowledge sharing.

In both areas, large amounts of data flow from users and get synthesized and used to affect the virtual and/or real world. These applications raise new challenges, including how to process heterogeneous data streams in order to materialize real-world events, how to balance privacy against the collective benefit of sharing personal real-time information, and how to apply more intelligent processing to send interesting events in the co-space to someone in the physical world.

The programming of virtual actors in games and virtual worlds requires large-scale parallel programming; declarative methods have been proposed as a solution in this environment, as discussed earlier. These applications also require development of efficient systems, as suggested earlier in the context of database engines, including appropriate storage and retrieval methods, data-processing engines, parallel and distributed architectures, and power-sensitive software techniques for managing the events and communications across large number of concurrent users.

### Moving Forward

The 2008 Claremont meeting also involved discussions on the database research community’s processes, including organization of publication procedures, research agendas, attraction and mentorship of new talent, and efforts to ensure a benefit from the research on practice and toward furthering our understanding of the field. Some of the trends seen in database research are echoed in other areas of computer science. Whether or not they are, the discussion may be of broader interest in the field.

Prior to the meeting, a team led by one of the participants performed a bit of ad hoc data analysis over database conference bibliographies from the DBLP repository (dblp.uni-trier.de). While the effort was not scientific, the results indicated that the database research community has doubled in size over the past decade, as suggested by several metrics: number of published papers, number of distinct authors, number of distinct institutions to which these authors belong, and number of session topics at conferences, loosely defined. This served as a backdrop to the discussion that followed. An open question is whether this phenomenon is emerging at larger scales—in computer science and in science in general. If so, it may be useful to discuss the management of growth at those larger scales.

The growth of the database community puts pressure on the content and processes of database research publications. In terms of content, the increasingly technical scope of the community makes it difficult for individual researchers to keep track of the field. As a result, survey articles and tutorials are increasingly important to the community. These efforts should be encouraged informally within the community, as well as via professional incentive structures (such as academic tenure and promotion in industrial labs). In terms of processes, the reviewing load for papers is increasingly burdensome, and there was a perception at the Claremont meeting that the quality of reviews had been decreasing. It was suggested at the meeting that the lack of face-to-face program-committee meetings in recent years has exacerbated the problem of poor reviews and removed opportunities for risky or speculative papers to be championed effectively over well-executed but more pedestrian work.

There was some discussion at the meeting about recent efforts—notably by ACM-SIGMOD and VLDB—to enhance the professionalism of papers and the reviewing process via such mechanisms as double-blind reviewing and techniques to encourage experimental repeatability. Many participants were skeptical that the efforts to date have contributed to long-term research quality, as measured in

intellectual and practical relevance. At the same time, it was acknowledged that the database community's growth increases the need for clear and clearly enforced processes for scientific publication. The challenge going forward is to find policies that simultaneously reward big ideas and risk-taking while providing clear and fair rules for achieving these rewards. The publication venues would do well to focus as much energy on processes to encourage relevance and innovation as they do on processes to encourage rigor and discipline.

In addition to tuning the mainstream publication venues, there is an opportunity to take advantage of other channels of communication. For example, the database research community has had little presence in the relatively active market for technical books. Given the growing population of developers working with big data sets, there is a need for accessible books on scalable data-management algorithms and techniques that programmers can use to build software. The current crop of college textbooks is not targeted at this market. There is also an opportunity to present database research contributions as big ideas in their own right, targeted at intellectually curious readers outside the specialty. In addition to books, electronic media (such as blogs and wikis) can complement technical papers by opening up different stages of the research life cycle to discussion, including status reports on ongoing projects, concise presentation of big ideas, vision statements, and speculation. Online fora can also spur debate and discussion if appropriately provocative. Electronic media underscore the modern reality that it is easy to be widely published but much more difficult to be widely read. This point should be reflected in the mainstream publication context, as well as by authors and reviewers. In the end, the consumers of an idea define its value.

Given the growth in the database research community, the time is ripe for ambitious projects to stimulate collaboration and cross-fertilization of ideas. One proposal is to foster more data-driven research by building a globally shared collection of structured data, accepting contributions

from all parties. Unlike previous efforts in this vein, the collection should not be designed for any particular benchmark; in fact, it is likely that most of the interesting problems suggested by this data are as yet unidentified.

There was also discussion at the meeting of the role of open source software development in the database community. Despite a tradition of open source software, academic database researchers have only rarely reused or shared software. Given the current climate, it might be useful to move more aggressively toward sharing software and collaborating on software projects across institutions. Information integration was mentioned as an area in which such an effort is emerging.

Finally, interest was expressed in technical competitions akin to the Netflix Prize ([www.netflixprize.com](http://www.netflixprize.com)) and KDD Cup ([www.sigkdd.org/kddcup/index.php](http://www.sigkdd.org/kddcup/index.php)) competitions. To kick off this effort in the database domain, meeting participants identified two promising areas for competitions: system components for cloud computing (likely measured in terms of efficiency) and large-scale information extraction (likely measured in terms of accuracy and efficiency). While it was noted that each of these proposals requires a great deal of time and care to realize, several participants volunteered to initiate efforts. That work has begun with the 2009 SIGMOD Programming Contest ([db.csail.mit.edu/sigmod09contest](http://db.csail.mit.edu/sigmod09contest)). □

#### References

1. Abiteboul, S. et al. The Lowell database research self assessment. *Commun. ACM* 48, 5 (May 2005), 111–118.
2. Austin, I. I.B.M. acquires Cognos, maker of business software, for \$4.9 billion. *New York Times* (Nov. 11, 2007).
3. Bernstein, P.A. et al. The Asilomar report on database research. *SIGMOD Record* 27, 4 (Dec. 1998), 74–80.
4. Bernstein, P.A. et al. Future directions in DBMS research: The Laguna Beach participants. *SIGMOD Record* 18, 1 (Mar. 1989), 17–26.
5. Silberschatz, A. and Zdonik, S. Strategic directions in database systems: Breaking out of the box. *ACM Computing Surveys* 28, 4 (Dec. 1996), 764–778.
6. Silberschatz, A., Stonebraker, M., and Ullman, J.D. Database research: Achievements and opportunities into the 21st century. *SIGMOD Record* 25, 1 (Mar. 1996), 52–63.
7. Silberschatz, A., Stonebraker, M., and Ullman, J.D. Database systems: Achievements and opportunities. *Commun. ACM* 34, 10 (Oct. 1991), 110–120.

Correspondence regarding this article should be addressed to Joseph M. Hellerstein (hellerstein@cs.berkeley.edu).

© 2009 ACM 0001-0782/09/0600 \$10.00

# contributed articles

DOI:10.1145/1516046.1516063

**The vision is being overwhelmed by the reality of business, politics, logistics, and competing interests worldwide.**

BY KENNETH L. KRAEMER, JASON DEDRICK, AND PRAKUL SHARMA

## One Laptop Per Child: Vision vs. Reality

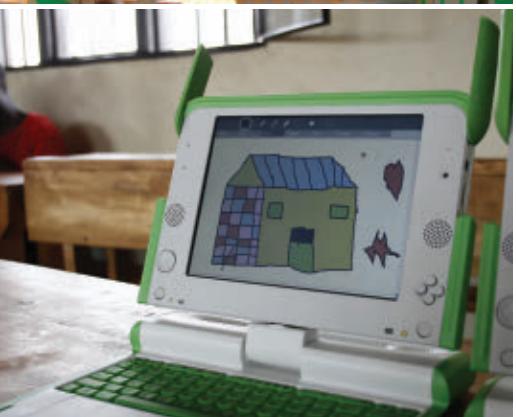
AT THE WORLD Economic Forum in Davos, Switzerland, January 2005, Nicholas Negroponte unveiled the idea of One Laptop Per Child (OLPC), a \$100 PC that would transform education for the world's disadvantaged schoolchildren by giving them the means to teach themselves and each other. He estimated that up to 150 million of these laptops could be shipped annually by the end of 2007.<sup>4</sup> With \$20 million in startup investment, sponsorships and partnerships with major IT industry players, and interest from developing countries, the nonprofit OLPC project generated excitement among international leaders and the world media. Yet as of June 2009 only a few hundred thousand laptops have been distributed (they were first available in 2007), and OLPC has been forced to dramatically scale back its ambitions.

Although some developing countries are indeed deploying OLPC laptops, others have cancelled planned deployments or are waiting on the results of pilot projects before deciding whether to acquire them in numbers. Meanwhile, the OLPC organization ([www.olpc.com/](http://www.olpc.com/)) struggles with key staff defections, budget cuts, and ideological disillusionment, as it appears to some that the educational mission has given way to just getting laptops out the door. In addition, low-cost commercial netbooks from Acer, Asus, Hewlett-Packard, and other PC vendors have been launched with great early success.

So rather than distributing millions of laptops to poor children itself, OLPC has motivated the PC industry to develop lower-cost, education-oriented PCs, providing developing countries with low-cost computing options directly in competition with OLPC's own innovation. In that sense, OLPC's apparent failure may be a step toward broader success in providing a new tool for children in developing countries. However, it is also clear that the PC industry cannot profitably reach millions of the poorest children, so the OLPC objectives might never be achieved through the commercial market alone.

Here, we review and analyze the OLPC experience, focusing on the two most important issues: the successes and failures of OLPC in understanding and adapting to the developing-country environment and the unexpectedly aggressive reaction by the PC industry, including superpowers Intel and Microsoft, to defeat or co-opt the OLPC effort.

OLPC created a novel technology, the XO laptop, developed with close attention to the needs of students in poor rural areas. Yet it failed to anticipate the social and institutional problems that could arise in trying to diffuse that innovation in the developing-country context. In addition, OLPC has been stymied by underestimating the aggressive reaction of the PC industry to the perceived threat of a \$100 laptop



## Worldwide distribution of XO laptops.



Country	OLPC Web site <sup>a</sup>	Actual Deployments	Date of Actual Deployment Information/Detail
Uruguay	202,000	150,000	November 2008 <sup>b</sup>
Peru	145,000	40,000	100,000 in distribution <sup>c</sup>
Mexico	50,000	50,000	Starting to be shipped <sup>d</sup>
Haiti	13,000	Dozens	Pilot began in summer 2008 <sup>e</sup>
Afghanistan	11,000	450	Expected to rise to 2010 <sup>f</sup>
Mongolia	10,100	3,000	G1G1 laptops beneficiary <sup>g</sup>
Rwanda	16,000	10,000	Arrived, not deployed; infrastructure issues <sup>h</sup>
Nepal	6,000	6,000	Delivered April 2007 <sup>i</sup>
Ethiopia	5,000	5,000	Three schools <sup>j</sup>
Paraguay	4,000	150	4,000 planned next quarter <sup>k</sup>
Cambodia	3,200	1,040	January 29, 2009 <sup>l</sup>
Guatemala	3,000	—	Planned before third quarter 2009 <sup>m</sup>
Colombia	2,600	1,580	January 25, 2009 <sup>n</sup> ; agreement to buy 65,000 XO <sup>o</sup>
Brazil	2,600	630	February 6, 2009 <sup>p</sup>
India	505	31	January 20, 2009 <sup>q</sup>

**a** OLPC numbers include "XO's delivered, shipped, or ordered" but do not distinguish between these categories; [wiki.laptop.org/go/Deployments](http://wiki.laptop.org/go/Deployments)

**b** Tabare, V. Uruguay: When education meets technology. *Miami Herald* (Nov. 22, 2008), A21.

**c** Peru on the up and up, lessons to be learned. *Business News Americas* (Dec. 18, 2008).

**d** [www.bniamericas.com/story.xls?&id\\_sector=1&id\\_noticia=431002&Tx\\_idioma=I&source=1](http://www.bniamericas.com/story.xls?&id_sector=1&id_noticia=431002&Tx_idioma=I&source=1)

**e** [www.olpc.eu.org/content/xo\\_stories/haiti/Haiti.html](http://www.olpc.eu.org/content/xo_stories/haiti/Haiti.html)

**f** [www.olpcnews.com/countries/afghanistan/olpc\\_afghanistan\\_first\\_school\\_day.html](http://www.olpcnews.com/countries/afghanistan/olpc_afghanistan_first_school_day.html)

**g** [www.olpc.eu.org/content/xo\\_stories/mongolia/Mongolia.html](http://www.olpc.eu.org/content/xo_stories/mongolia/Mongolia.html)

**h** [www.olpc.eu.org/content/xo\\_stories/rwanda/Rwanda.html](http://www.olpc.eu.org/content/xo_stories/rwanda/Rwanda.html)

**i** [www.olpc.eu.org/content/xo\\_stories/nepal/Nepal.html](http://www.olpc.eu.org/content/xo_stories/nepal/Nepal.html)

**j** [http://www.olpc.eu.org/content/xo\\_stories/ethiopia/Ethiopia.html](http://www.olpc.eu.org/content/xo_stories/ethiopia/Ethiopia.html)

**k** Bucaramanga computers, OLPC, Gemalto. *Business News Americas* (Feb. 9, 2009).

**l** [wiki.laptop.org/go/OLPC\\_Cambodia](http://wiki.laptop.org/go/OLPC_Cambodia)

**m** [wiki.laptop.org/go/OLPC\\_Guatemala](http://wiki.laptop.org/go/OLPC_Guatemala)

**n** [wiki.laptop.org/go/OLPC\\_Colombia](http://wiki.laptop.org/go/OLPC_Colombia)

**o** Pilar Saenz, OLPC Volunteer in Colombia (email)

**p** [download.laptop.org/content/conf/20080520-country-wkshp/Presentations/OLPC%20Country%20Meeting%20-%20Day%204%20-%20May%2023rd,%202008/Brazil%20-%20Jose%20Aquino%20-%20Govt%20of%20Brazil.ppt#266,8,Slide 8](http://download.laptop.org/content/conf/20080520-country-wkshp/Presentations/OLPC%20Country%20Meeting%20-%20Day%204%20-%20May%2023rd,%202008/Brazil%20-%20Jose%20Aquino%20-%20Govt%20of%20Brazil.ppt#266,8,Slide 8)

**q** [www.olpc.eu.org/content/xo\\_stories/india/India.html](http://www.olpc.eu.org/content/xo_stories/india/India.html)

being widely distributed in places the industry sees as emerging markets for its own products.

The case of OLPC can be seen as a study in the general diffusion of innovation in developing countries. Our analysis draws on diffusion-of-innovation theory, exemplified by Rogers,<sup>18</sup> and illustrates the difficulty in getting widespread adoption of even proven innovation due to misunderstanding the social and cultural environment in which the innovation is to be introduced. We also bring to bear specific insights from the literature on adoption of IT in developing countries,<sup>2,25</sup> using them to analyze the OLPC experience and draw implications for developers and policymakers.

The original OLPC vision was to change education through the development and distribution of low-cost laptops embodying a new learning model to every child in the developing countries. Despite shifting over time, it can be characterized by the following text from the OLPC charter: "OLPC is not, at heart, a technology program, nor is the XO a product in any conventional sense of the word. OLPC is a nonprofit organization providing a means to an end—an end that sees children in even the most remote regions of the globe being given the opportunity to tap into their own potential, to be exposed to a whole world of ideas, and to contribute to a more productive and saner world community" ([www.olpcnews.com/people/negroponte/new.olpc.mission.statement.html](http://www.olpcnews.com/people/negroponte/new.olpc.mission.statement.html)).

Conceived and led by Nicholas Negroponte, a former director of MIT's Media Lab, OLPC aimed to achieve its vision through extraordinary innovation in hardware and software that fosters self-learning and fits with the often-harsh environment in developing countries. The hardware was to be a \$100 laptop that would make affordable the large-scale deployment of computer networks in their schools.

The XO laptop developed by OLPC reflects hardware innovation in the power supply, display, networking, keyboard, and touchpad to provide a durable and interactive laptop (see the figure here). The shell of the machine is resistant to dirt and moisture, with all key parts designed to fit behind the display. It contains a pivoting, reversible,

dual-mode (monochrome for outside, color for indoors) display, movable rubber WiFi antennas with wireless mesh networking, and a sealed rubber-membrane keyboard that can be customized for different languages. For low power consumption and ruggedness, the XO design intentionally omits all motor-driven moving parts. It was developed jointly by the MIT Media Lab, OLPC, and Quanta, a Taiwan-based original design manufacturer, and is manufactured by Quanta in Songjiang, China.

The software for the XO consists of a pared-down version of the Fedora Linux operating system and specially designed graphical user interface called Sugar. It was developed by the project to explore naturalistic concepts related to learning, openness, and collaboration.<sup>a</sup>

#### Pilot Implementation

High-level officials, including even prime ministers and education ministers, in some developing countries are enthusiastic about OLPC, committed to purchases and/or trial-distribution projects. OLPC pilots in a half-dozen countries report positive changes (such as increased enrollment in schools, decreased absenteeism, increased discipline, and more participation in classrooms), but it is not clear if these changes are directly related to OLPC, as many evaluations are neither independent nor systematic. Independent evaluations in Ethiopia and Uruguay cite a positive effect on the availability of learning material via the laptop but also problems with buggy input devices, connectivity, software functionality, and teacher training.<sup>8,12,13</sup>

As of June 2009 the largest ongoing pilot project is in Peru, which planned to distribute 140,000 XOs in 2008, even into rural areas high in the Andes where electricity is often limited and Internet connections are not available. There is enthusiasm among students and teach-

<sup>a</sup> Chief among them are collaboration and expression (such as Web browsing, email, online chat, word processing, drawing, music sequencing, and programming); groups and neighborhoods to signify other users in physical and logical proximity; a view-source-code key to encourage users to tinker with the code; replacing files and folders with “journals” that store activities performed by users; and tagging, clipping, sharing, and searching as systemwide features.<sup>22</sup>

## Expecting a laptop to cause such revolutionary change showed a degree of naiveté, even for an organization with the best intentions and smartest people.

ers in the villages and support from the national education ministry and regional governors who have requested 500,000 more laptops.<sup>9</sup> However, reports from the classroom suggest that teacher training is limited, and willingness to adopt a new approach to teaching is questionable. Children are excited but somewhat confused about the use of the machines, and educational software is lacking or difficult to use. Also, if a machine fails, it is up to the family to replace it or the child must do without.<sup>20</sup>

#### Targeted Cost

Despite its considerable innovation, or perhaps because of it, the OLPC project has been unable to achieve its \$100 targeted cost. The current cost of each unit is listed on the OLPC Website as \$199 ([www.laptop.org/en/participate/ways-to-give.shtml](http://www.laptop.org/en/participate/ways-to-give.shtml)). However, this does not include upfront deployment costs, which are said to add an additional 5%–10% to the cost of each machine ([wiki.laptop.org/go/Larger\\_OLPC](http://wiki.laptop.org/go/Larger_OLPC)), and subsequent IT-management costs. Nor does it include the cost of teacher training, additional software, and ongoing maintenance and support. OLPC initially required governments to purchase a million units, then reduced the number to 250,000 in April 2007. Such large purchases are difficult to justify for governments in developing countries, and the requirement was ultimately eliminated.

Some countries eventually lost interest due to the higher costs of the XO. For example, Nigeria failed to honor a pledge by its former president to purchase a million units, partly because they no longer cost \$100 apiece.<sup>21</sup> Meanwhile, other countries, including Libya, have opted for the Intel Classmate, which is priced at approximately \$250 for the PC alone. Officials in Libya, which had planned to buy up to 1.2 million XO laptops, became concerned that the machines lacked Windows, and that service, teacher training, and future upgrades would not be provided directly by OLPC. Subsidies from Intel, including donated laptops and teacher training, also helped persuade the Libyan government to choose the Classmate.<sup>21</sup>

#### Production, Sales, Distribution

OLPC originally estimated that it would

ship 100–150 million XO laptops by the end of 2007, but the program has clearly fallen far short. Under more modest goals, production was supposed to reach five million laptops by the end of 2008. By contrast, industry analysts report that Quanta's manufacturing effort began only in December 2007 and reached a total of 370,500 units by third quarter 2008.<sup>16</sup>

Early commitments for a million XOs each from Brazil, Libya, and Nigeria evaporated, but relatively large purchases were made by Uruguay (200,000), Peru (145,000), and Mexico (50,000). In November 2007, OLPC launched a philanthropy program called Give One Get One (G1G1, [www.olpcnews.com/countries/usa/olpc\\_xo\\_laptop\\_sale.html](http://www.olpcnews.com/countries/usa/olpc_xo_laptop_sale.html)) where people in the U.S. could buy two machines for \$399, with one being sent to a child in a developing country. The first program was successful, with about 167,000 units sold, but a second G1G1 program in November 2008 resulted in only 12,500 units sold.

Lagging production and sales mean that distribution has also lagged. The table here lists distribution as reported by OLPC, but many units have yet to be deployed to their intended recipients. What has the project accomplished? Why is it so short of its original goals? To answer, we look in more detail at where OLPC succeeded and failed in understanding the developing-country environment and how it was being confronted by the PC industry.

### **Analysis**

OLPC dedicated a great deal of effort to designing a laptop that would function well in a developing-country environment. OLPC's technologist culture encouraged innovation, showing a good understanding of what was needed in developing countries. For example, the XO is sealed to keep out dirt, has a display that can be read in bright sunlight, runs on low power, and is rugged.

At the same time, the decision to use the Linux/Sugar operating system and interface was driven by a combination of pragmatic considerations and open source ideology. From a pragmatic point of view, Linux doesn't require the computing power of Windows and has a price tag (zero) compatible with the goal of minimizing cost.

Diffusion of IT innovation does not

**PC makers across the board are still seeking a formula for well-designed, low-cost computing devices, along with a complementary delivery value chain, market strategy, and business model.**

depend only on the nature of the innovation itself. Often, more important is the social and cultural environment in which it will operate.<sup>3,26</sup> Information technologies are not standalone innovations but system innovations, the value of which depends largely on an ecosystem that includes hardware, applications, peripherals, network infrastructure, and services (such as installation, training, repair, and technical support). Deployment involves training teachers, creating software and digital content, delivering maintenance and support, and sustaining a long-term commitment. Such capabilities are in short supply in developing countries,<sup>7,26</sup> and OLPC simply never had the resources to provide them.

The OLPC plan was to rely on governments to buy its machines, provide distribution and support, train teachers to use and maintain them, and even sponsor development of local-language software. OLPC established its own distribution network or worked with local voluntary organizations in some countries to help with implementation. For global distribution, OLPC reached (in 2007) a comprehensive agreement with cellphone distributor Brightstar of Miami, FL, to help manage the complexities of entering diverse markets.<sup>23</sup> However, none of these institutions had the ability to scale up to deployment of millions of machines. This situation is common in developing countries where endemic problems of infrastructure, financial resources, technical skills, and waning political support "hinder both the completion of IS innovation initiatives and the realization of their expected benefits."<sup>b</sup>

IT innovation is also part of socially embedded systems, the use of which cannot be isolated from the social and cultural environment or from local norms of practice.<sup>1,25</sup> In some cases, teachers and the educational establishment have resisted innovation that

<sup>b</sup> Negroponte seems to question whether teachers are needed at all. Speaking about providing the rural poor a solid educational basis for development at the 2007 Digital, Life, Design conference in Munich, Germany, Negroponte said: "It's not about training teachers. It's not about building schools. With all due respect [to Hewlett-Packard's e-inclusion efforts], it's not about curriculum or content. It's about leveraging the children themselves."<sup>24</sup>

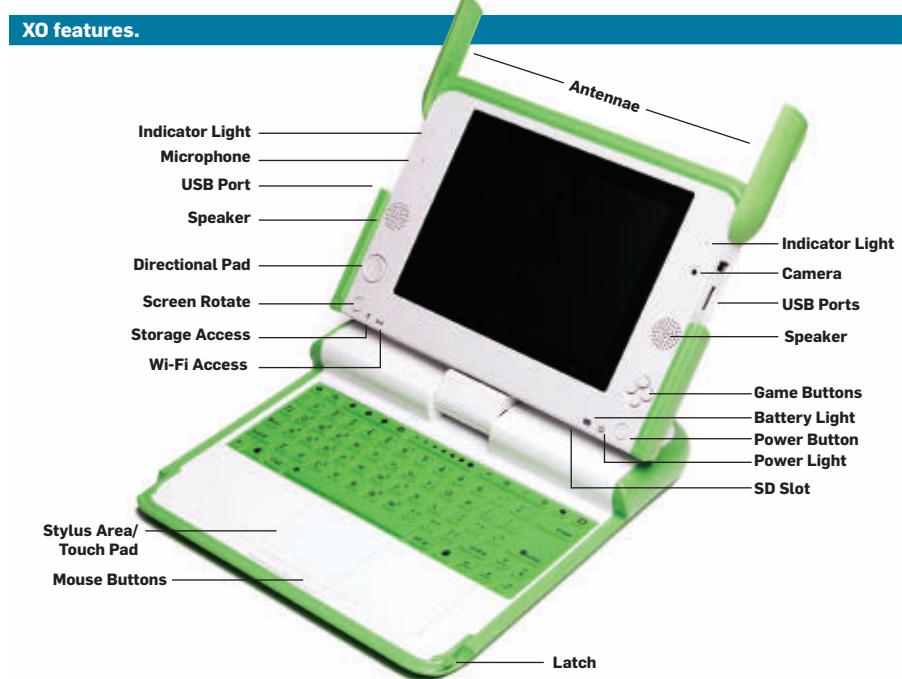
requires a significant change in pedagogy and that might reduce teacher status.<sup>b</sup> Even when the laptops are adopted, they are not always used as envisioned by OLPC or by education ministers. One Peruvian teacher said, "The ministry would want us to use the laptop every day for long periods of time. But we have decided to set rules in our school and, really, the laptop, it's only a tool for us."<sup>10</sup>

Such resistance is no surprise to students of innovation diffusion or of IT for development. Rogers<sup>18</sup> pointed to examples where innovation diffusion failed due to cultural norms and the effects of such innovation on existing institutional arrangements. Avgerou<sup>2</sup> noted that attitudes toward hierarchy are particularly problematic in developing countries. An example illustrating both themes is that the Peruvian experiment was initiated without being explained to the national teachers' union.<sup>10</sup> OLPC has strong support from the Peruvian Education Ministry, but ultimately teachers must actually use the machines in the classroom, and they are likely to see the union as an ally while possibly mistrusting the ministry.

The fact that OLPC was much stronger in developing innovative technology than in understanding how to diffuse it may reflect the engineering orientation of the organization and its lack of understanding of the needs or interests of the nontechnical people who will ultimately buy and use the innovation. This is illustrated by David Cavallo, OLPC's chief education architect, saying, "We're hoping that these countries won't just make up ground but will jump into a new educational environment."<sup>9</sup> Expecting a laptop to cause such revolutionary change showed a degree of naiveté, even for an organization with the best intentions and smartest people.

### Competitive Response from the PC Industry

The OLPC project was a potential threat to the PC industry in emerging markets. OLPC's use of an AMD microprocessor and Linux operating system was a potential threat to the dominant position and historically high profit margins of Intel and Microsoft. Its targeting of a new market (developing-country



schools) that existing PC makers were not serving raised the prospect that OLPC might gain a foothold in emerging markets more generally. Moreover, the XO's ultra-low price raised the likelihood of a new price point for notebooks, potentially forcing PC makers to cannibalize existing low-end products in order to compete (and is what ultimately happened).

Branded PC makers have always faced competition from cheap local brands and clone makers in developing countries, but OLPC threatened to grab a share of education budgets worldwide that PC makers hoped to tap for themselves. Negroponte's high-profile announcement of the project and the publicity he garnered quickly caught the industry's attention.

Leading companies first responded by disparaging the XO as a useless toy. Intel's Craig Barrett called it "a gadget," saying people want the full functionality of a PC.<sup>17</sup> Bill Gates said "...geez, get a decent computer where you can actually read the text and you're not sitting there cranking the thing while you're trying to type."<sup>11</sup> Before long, however, the industry began to respond with action, not just words.

In 2006, Intel introduced a small laptop—the Classmate—for developing countries that today sells for \$230–\$300. Intel has since licensed the Classmate reference design to PC makers to manufacture and distribute and

is marketing it aggressively against the XO worldwide. It secured deals to sell hundreds of thousands of Classmates in Libya, Nigeria, and Pakistan, some of the very countries OLPC was counting on. Intel launched a series of pilot projects in these countries, saying it will also test the Classmate in at least 22 others while donating thousands of machines.<sup>21</sup> Intel briefly joined OLPC in July 2007 but got into a nondisparagement dispute with Negroponte and dropped out only seven months later.<sup>14</sup>

In 2007, Microsoft offered to make available Windows, a student version of Microsoft Office, and educational programs to developing countries for \$3 per copy when used on computers in schools. OLPC then decided to allow Windows on the XO, a choice driven by demands from some governments for Windows-based PCs. Even in countries with very low levels of PC penetration, officials who make purchasing decisions may favor a technology standard (the Wintel design) they are familiar with or believe children must learn on systems they will encounter later in the work force.

The OLPC project also stimulated innovation in low-cost, low-power PCs. Seeing OLPC's success in developing a sub-\$200 notebook, Asustek introduced the EeePC notebook in 2007 for the educational and consumer markets in both developed and developing countries, selling more than 300,000



**The 2010 version of the One Laptop per Child, the XO-2, will have a foldable e-book form and reduce power consumption to one watt.**

units in four months. It was soon joined by major PC makers, including Acer, Dell, Hewlett-Packard, and many smaller ones in creating a new category of PC known today as netbooks.

While the XO was specifically designed for the poor, rural education market in developing countries, netbook vendors target urban consumer and education markets in developed, as well as emerging, markets. In 2008, the netbook market exploded, with sales of 10 million units worldwide mostly running Intel's low-cost Atom processor and Windows; sales are expected to double in 2009.<sup>16</sup>

The OLPC has been credited with spurring the netbook market, but the competition it spurred is now OLPC's own biggest challenge. Developing countries today have a wide choice of vendors offering inexpensive netbooks, and, though not designed like the XO for the rigors of poor rural villages, they are competitive in large, easier-to-serve urban populations. OLPC responded by announcing in January 2009 that its second-generation laptop design would be licensed freely to PC makers to manufacture and distribute, hoping to use the resources of these firms to get millions of laptops into the hands of poor children in developing countries. Negroponte hopes to have a prototype in 18 months (from January),

selling them for perhaps \$75 each.<sup>5</sup>

### Lessons

The OLPC experience offers lessons for innovators and others aiming to introduce and deploy IT innovation to benefit the poor, as well as for the governments of developing countries. For innovators, we thus draw three general lessons:

*Diffusing a new innovation requires understanding the local environment.* OLPC recognized correctly that laptops could reach the poorest children only if they were subsidized by government or other funding sources. This is similar to rural electrification and telephone service, which usually cannot be provided economically and end up subsidized by government or by charges to urban customers who can be served profitably. However, innovators should understand that governments are not monolithic entities, nor are they the same from one country to the next. In some cases, funding can be allocated by an education ministry, in others it must be approved by the legislature, and in others provincial or local governments have jurisdiction. Commitments from high-level officials or political leaders are as binding as a politician's campaign promises.<sup>26</sup> Flying into a country and winning initial support is only a first step and must be followed by a sustained effort by people

with a deep understanding of the local environment to ensure commitment leads to money and action.

Likewise, social, economic, and cultural environments vary greatly across and even within countries, and deploying new technologies requires understanding these environments. Innovators must consider the need for expertise in sociology, anthropology, public policy, and economics, as well as for engineers, and establish coherent criteria for selecting countries to target based on social, economic, and political characteristics. Success in a few developing countries is critical to broad diffusion, as potential adopters look to their peers for evidence of the value of the innovation.<sup>18</sup>

*Innovative technology can be disruptive and trigger a backlash from incumbents.* Some innovations pose a threat to industry incumbents, who may seek to undermine the innovator's efforts. The more visible the threat, the stronger the reaction is likely to be. This illustrates a dilemma for developers. A program less ambitious and less publicized than OLPC might not attract the attention of industry incumbents but also might not attract the partners, investors, and other sponsors needed to develop and deploy the innovation. As multinational companies direct more attention to emerging markets and so-called "bottom of the pyramid" consumers, there is more likelihood of competition but also more opportunity for cooperation as well. PC makers across the board are still seeking a formula for well-designed, low-cost computing devices, along with a complementary delivery value chain, market strategy, and business model.

*Innovative information technologies do not stand alone.* A technology like the XO is a system-level innovation that requires complementary assets to be valuable. While OLPC was able to deliver high-level design and hand off development and manufacturing to Quanta, it had no one to handle marketing, deployment, and support.<sup>15</sup> Unlike the commercial PC companies, it was not part of any established business ecology and lacked resources to establish its own ecology.

For developing countries, international agencies, and philanthropists, there are other kinds of lessons:

*Understand the true costs and risks, as well as benefits, of innovation.* IT innovation like the XO may offer great benefits but also involves costs and risks. The purchase of a laptop is merely the start of a stream of ongoing costs. The total cost of ownership for a laptop program could include infrastructure investment, training, tech support, hardware maintenance, software licenses and upgrades, and replacement expenditures. Cost can also include the opportunity cost or the foregone investment in teachers, facilities, or other educational materials<sup>7</sup> cited by India's education ministry as its main reason for not joining OLPC.<sup>6</sup>

There is also a risk that the expected benefits might not be realized. Problems in implementation could limit actual use, and the need for ongoing funding means that the innovation might not be sustainable beyond some initial period.<sup>2,13</sup> Another risk is investing in a technology platform that might not be supported in the future; for instance, investment in software, content, and training for the XO platform could be wasted if OLPC would disappear.

Policymakers are able to reduce the risk if they make major acquisition decisions only after careful evaluation of pilot projects that enable learning firsthand how the technology fits with their educational goals and environment. Learning from other countries' experience can be valuable even when the context is different; Al-Gahtani<sup>1</sup> says that successful pilot projects by peers in other developing countries help reduce the perceived risk of adoption.

*Adopting organizations need to develop internal capabilities and set priorities.* Although governments might receive outside assistance for trials, they must be able to sustain the innovation in the development of digital educational content, training of teachers to integrate ICT-based educational materials in the teaching-learning process, and design and installation of supporting IT and power infrastructure. For example, one independent evaluation concluded: "While the Uruguayan government is making a great effort in providing funding for the hardware, there is no funding for designing and developing software and content for use with the laptops or for conducting a thorough evaluation of the edu-

cational and societal outcomes of the project."<sup>13</sup> Other evaluations argue that the countrywide deployments envisaged by OLPC are simply beyond the resources of any developing country, saying that governments must set priorities regarding goals and the regions, sectors, and schools to be served.<sup>8,12</sup>

## Conclusion

The potential significance of the XO, as well as of other IT innovations, in developing countries calls for systematic, independent evaluation—a true "grand challenge" for the computing and social science communities. Researchers can provide value by conducting well-designed studies of the diffusion and results of such innovation. The knowledge created promises to prevent wasting a great deal of money and effort and lead to quicker diffusion and better use of innovations that prove beneficial. While OLPC has so far fallen short of its goals, there is much yet to be learned by studying this case of IT innovation.

## Acknowledgments

The Personal Computing Industry Center ([pcic.merage.uci.edu/](http://pcic.merage.uci.edu/)) is supported by grants from the Alfred P. Sloan Foundation and the U.S. National Science Foundation. Any opinions, findings, and conclusions or recommendations expressed in this article are those of the author(s) and do not necessarily reflect the views of the Sloan Foundation or the National Science Foundation. □

## References

1. Al-Gahtani, S.S. Computer technology adoption in Saudi Arabia: Correlates of perceived innovation attributes. *Information Technology for Development* 10, 1 (Jan. 2003), 57–69.
2. Avgerou, C. Information systems in developing countries: A critical research review. *Journal of Information Technology* 23 (June 2008), 133–146.
3. Avgerou, C. The significance of context in information systems and organizational change. *Information Systems Journal* 11, 1 (January 2001), 43–63.
4. BBC News. Sub-\$100 laptop design unveiled (Sept. 29, 2005); [news.bbc.co.uk/2/hi/technology/4292854.stm](http://news.bbc.co.uk/2/hi/technology/4292854.stm).
5. Bray, H. Cheaper laptop promised; Negroponte remains determined to realize vision. *Boston Globe* (Feb. 11, 2009); [www.boston.com/business/technology/articles/2009/02/11/cheaper Cheap\\_laptop\\_promised/](http://www.boston.com/business/technology/articles/2009/02/11/cheaper Cheap_laptop_promised/).
6. Einhorn, B. A crusade to connect children. India criticizes an MIT professor's quest to provide 'One Laptop Per Child'; but he's forging ahead elsewhere. *BusinessWeek.com* (Aug. 16, 2006); [www.businessweek.com/globalbiz/content/aug2006/gb20060816\\_021986.htm](http://www.businessweek.com/globalbiz/content/aug2006/gb20060816_021986.htm).
7. Farrell, G. ICT in education in Rwanda. In *Survey of ICD and Education in Africa: Rwanda Country Report*. World Bank Information Development, Washington D.C., Dec. 2007; [www.infodev.org/en/Publication.423.htm](http://www.infodev.org/en/Publication.423.htm).
8. Haertl, H. *Low-Cost Devices in Educational Systems: The Use of the 'XO-Laptop' in the Ethiopian Educational System*. Report distributed by the Division of Health, Education and Social Protection, Information and Communication Technologies, GTZ-Project, Deutsche Gesellschaft für Technische Zusammenarbeit, Eschborn, Germany, Jan. 2008; [www.gtz.de/de/dokumente/gtz/2008-en-laptop.pdf](http://www.gtz.de/de/dokumente/gtz/2008-en-laptop.pdf).
9. Hamm, S., Smith, G., and Lakshman, N. Social cause meets business reality. *BusinessWeek in Focus* (June 12, 2008), 48; [www.thefreelibrary.com/cial+Cause+Meets+Business+Reality-a01611563648](http://www.thefreelibrary.com/cial+Cause+Meets+Business+Reality-a01611563648).
10. Hansen, L. Laptop deal links rural Peru to opportunity, risk. *Weekend Edition*. National Public Radio (Sunday, Dec. 14, 2008).
11. Hiser, S. Bill Gates criticises OLPC. *PlexNex blog* (Mar. 16, 2006); [fussnotes.typepad.com/plexnex/2006/03/bill\\_gates\\_crit.html](http://fussnotes.typepad.com/plexnex/2006/03/bill_gates_crit.html).
12. Hooker, M. *1:1 Technologies/Computing in the Developing World: Challenging the Digital Divide*. Global e-Schools and Communities Initiative, Dublin, Ireland, May 2008; [www.gesci.org/index.php?option=com\\_content&task=view&id=75&Itemid=64](http://www.gesci.org/index.php?option=com_content&task=view&id=75&Itemid=64).
13. Hourcade, J.P., Beittler, D., Cormenzana, F., and Flores, P. Early OLPC experiences in a rural Uruguayan school. In *Proceedings of CHI 2008* (Florence, Italy, Apr. 5–10). ACM Press, New York, 2008, 2503–2511.
14. Kirkpatrick, D. Negroponte on Intel's \$100 laptop pullout. *Fortune* (Jan. 4, 2008); [money.cnn.com/2008/01/04/technology/kirkpatrick\\_negroponte.fortune/index.htm](http://money.cnn.com/2008/01/04/technology/kirkpatrick_negroponte.fortune/index.htm).
15. Krstic, I. *Sic Transit Gloria Laptops*. Ivan Krstic blog (May 13, 2008); [radian.org/notebook/sic-transit-gloria-laptops](http://radian.org/notebook/sic-transit-gloria-laptops).
16. O'Donnell, B. *Worldwide Mini-Notebook PC 2008-2012 Forecast Update and 3Q08 Vendor Shares*. Market Analysis. IDC, Framingham, MA, Dec. 2008.
17. Reuters. Intel calls MIT's \$100 laptop a 'gadget'. *CNet news.com* (Dec. 9, 2005); [news.com/Intel+calls+MITs+100+laptop+a+gadget/t/2100-1005\\_3-5989067.html?tag=htnlAlert](http://news.com/Intel+calls+MITs+100+laptop+a+gadget/t/2100-1005_3-5989067.html?tag=htnlAlert).
18. Rogers, E.M. *Diffusion of Innovations, Fifth Edition*. Free Press, New York, 1995.
19. Shah, A. OLPC struggles to realize ambitious vision. *PC World* (Dec. 20, 2007); [www.pcworld.com/article/140698/olpc\\_struggles\\_to\\_realize\\_ambitious\\_vision.html](http://www.pcworld.com/article/140698/olpc_struggles_to_realize_ambitious_vision.html).
20. Simon, S. Laptops may change the way rural Peru learns. *Weekend Edition*. National Public Radio (Saturday, Dec. 13, 2008).
21. Stecklow, S. and Bandler, J. A little laptop with big ambitions. *WallStreetJournal.com* (Nov. 24, 2007); [online.wsj.com/public/article/SB119586754115002717.html](http://online.wsj.com/public/article/SB119586754115002717.html).
22. Vota, W. OMG: OLPC just gutted: 50% staff cut & more. *OLPC News Forum post* (Jan. 7, 2009); [www.olpcnews.com/forum/index.php?topic=4228.msg28414#msg28414](http://www.olpcnews.com/forum/index.php?topic=4228.msg28414#msg28414).
23. Vota, W. A Brightstar OLPC give one get one XO computer distributor. *One Laptop Per Child News* (Oct. 12, 2007); [www.olpcnews.com/implementation/plan/brightstar\\_xo\\_computer\\_distribution.html](http://www.olpcnews.com/implementation/plan/brightstar_xo_computer_distribution.html).
24. Vota, W. OLPC Nepal creates content while Negroponte dismisses it. *One Laptop Per Child News* (Jan. 31, 2007); [www.olpcnews.com/countries/nepal/negroponte\\_curriculum\\_content.html](http://www.olpcnews.com/countries/nepal/negroponte_curriculum_content.html).
25. Walshaw, G. and Sahay, S. Research on information systems in developing countries: Current landscape and future prospects. *Information Technology for Development* 12, 1 (Feb. 2006), 7–24.
26. Warschauer, M. Dissecting the 'digital divide': A case study in Egypt. *The Information Society* 19, 4 (Sept./Oct. 2003), 297–304.

**Kenneth L. Kraemer** (kkraemer@uci.edu) is a research professor in the Paul Merage School of Business, Co-Director of the Personal Computing Industry Center, and Associate Director of the Center for Research on Information Technology and Organizations, all at the University of California, Irvine.

**Jason Dedrick** (jdedrick@uci.edu) is Co-Director and a project scientist in the Personal Computing Industry Center at the University of California, Irvine.

**Prakul Sharma** (prakuls@uci.edu) is a research associate in the Personal Computing Industry Center at the University of California, Irvine.

# review articles

DOI:10.1145/1516046.1516064

**Information and communication technology for development can greatly improve quality of life for the world's neediest people.**

BY M. BERNARDINE DIAS AND ERIC BREWER

# How Computer Science Serves the Developing World

WHAT DO THE increasingly prominent news stories about \$100 laptops, kids learning about computers through a “hole in the wall,” and the power of mobile phones to educate, entertain, and connect people in remote regions have in common? It is the field of information and communication technology for development (ICTD), based on the belief that technology can have a large and positive effect on billions of individuals by helping them overcome the challenges so prevalent in developing regions. ICTD is not new—numerous important though relatively

low-profile projects have been building the foundations of ICTD for many years. What’s new are its name and, more important, the increased recognition the field has lately been receiving and its potential for exerting greater influence.

In this article we explore ICTD and examine the role that computer scientists can play in it. Our objective is to convince readers that although achieving all the goals of ICTD will not be easy, even their partial realization could have tremendous impact.

The motivation for this field comes from a new awakening to the vast gap in quality of life between the richest billion people on earth (who enjoy a variety of luxuries, including Internet access) and the poorest billion (who just barely eke out a living—and sometimes not). The base of the world’s economic pyramid has an estimated population of four billion—over half of our planet’s people—living on less than \$2 a day.

In response to this awakening, scholars and practitioners have begun to explore the transforming power of information and communication technology when applied to the problems traditionally addressed in development. Can mobile phones provide income generation and facilitate remote medical diagnosis? How can user interfaces be designed so they are accessible to the semiliterate and even the illiterate? What role can computers play in sustainable education for the rural poor? What new devices can we build to encourage literacy among visually impaired children living in poverty? What will a computer that is relevant and accessible to people in developing regions look like? These are just a few of the questions being addressed in ICTD.

In other words, ICTD can be seen as harnessing the power of information and communication technologies, or ICTs, to take up many of the challenges of development. ICTs include technologies ranging from robotic tools and state-of-the-art computers to desktop



**Educational initiatives by the TechBridgeWorld group at CMU explore the efficacy of technology tools like an automated English reading tutor. A more recent partnership with researchers from Ashesi University College in Ghana resulted in the country's first undergraduate robotics course.**

and laptop computers in their traditional forms; and from mobile phones, PDAs, and wireless networks to long-established technologies such as radio and television. The software components also span a wide range, from artificial intelligence and new algorithms, interfaces, and applications to the most prosaic programmed commodities.

Although the goals of international-development efforts vary, depending on the nature of each endeavor, the overarching goal of all such projects is the alleviation of the suffering caused by poverty and improvement of quality of life for the world's poor. The United Nations' eight Millennium Development Goals (MDGs) infused new energy into the world's development efforts and helped to focus them on concrete objectives—eradicating extreme poverty and hunger, improving maternal health, prevailing in the battle against HIV/AIDS and malaria, reducing child

mortality, and achieving universal primary education, environmental sustainability, and a global partnership for development—to be met by the year 2015. Other development goals, not emphasized in the MDGs, include access to adequate shelter, information, avenues for income generation, and financial credit. The ongoing rural-to-urban shift of so much of the world's population has introduced a new set of problems as well, including increased vulnerability to disasters and the corresponding challenges for effective disaster responses. These are among the many international-development challenges that ICTD researchers and practitioners hope to address. They expect to reinvent the form, function, and applications of ICTs in new and creative ways so that such challenges may best be met.

From a CS point of view, ICTD can be seen as the next wave in ubiquitous

computing. Historically, computers started as huge machines that filled rooms and were only relevant and accessible to a specialized minority. The next big wave was the home PC, which is now relevant and accessible to over one billion people worldwide. ICTD is perhaps the next revolution in computing—transforming the computer and the applications of computing so that this technology can finally become relevant and accessible to the other five billion people of the world.

Given its position at the intersection of technology and development, ICTD brings together a wide variety of actors in many different roles. Among the newest are computer scientists, and their role is potentially a big one, both for their beneficiaries and themselves. It can change the image of the computer science discipline, the nature of the PC, and the future of the field.

A crucial requirement for success

in ICTD, however, is interdisciplinary collaboration—working with scholars and practitioners from many different fields. Sociologists, ethnographers, and anthropologists, for example, can provide valuable information about the communities intended to benefit from ICTD. This information, regarding such things as cultural practices, traditions, languages, beliefs, and livelihoods, must guide the design and implementation processes for successful solutions in ICTD.

Economists and political scientists play important roles in ICTD as well by designing new economic models, marketing strategies, and governmental policies that affect the economic viability and sustainability of technological interventions. Social scientists also play a crucial role in evaluating the impacts and outcomes of ICTD projects using both qualitative and quantitative methods. They observe and predict how people in developing regions interact with technology, and they aim to affect social systems for adopting technology-aided solutions without disruption to the community. Thus computer scientists working in the field of ICTD must quickly learn to work with this variety of scholarly players, to benefit from their points of view, and to complement them wherever possible.

ICTD does not only cross disciplines; it also transcends the boundaries of academia and involves multiple sectors. This reality obliges ICTD researchers to work with practitioners, government representatives, multilateral institutions such as the United Nations, nonprofits, nongovernmental organizations, and even the private sector, whose interest in ICTD begins as it seeks access to emerging markets and new avenues for corporate social responsibility. Many of these sectors' people have been addressing the challenges of development for decades, and their efforts should profit from the addition of professionals in CS and related fields who will contribute new perspectives and their useful styles of rigor, critique, and innovation.

ICTD is therefore a truly global undertaking with a grand vision. It brings together numerous players, across geographic, socioeconomic, regional, disciplinary, and sectoral boundaries,

**Although the cause is noble and the impact can be large, ICTD must ultimately be judged on its research value—and in particular, its research value in computer science.**

who must work together if we are to improve the quality of life for the least privileged on our planet.

### The Many Challenges of ICTD

Given its enormous ambitions and multidisciplinary requirements, ICTD presents its researchers with a variety of challenges. They include adapting to unfamiliar cultures and traditions, ensuring accessibility to local languages and multiple levels of literacy, overcoming the barriers of misinformation and mistrust of technology, creating solutions that work within the local infrastructure, and many more. For example, networking must work in circumstances with low bandwidth, intermittent bandwidth, or no bandwidth at all. Computers must operate reliably in environments characterized by dust, heat, humidity, and inexperienced users. User interfaces must accommodate semiliterate and illiterate users. And software applications must be sufficiently intelligent to provide useful, accessible, and relevant services to populations that might be interacting with a computing system for the very first time.

Further, ICTD field tests often require considerable ingenuity, whether they involve accessing target communities, setting up long-term studies, transporting equipment, observing the logistics and legalities of export control laws, addressing safety concerns, and establishing trust and common ground with partnering organizations that cross cultural and geographic boundaries. And to begin with, researchers must be entrepreneurial in obtaining funding for their research, as ICTD is not yet an established field with reliable funding sources.

Although the cause is noble and the impact can be large, ICTD must ultimately be judged on its research value—and in particular, its research value in CS. Like other multidisciplinary fields, ICTD must be simultaneously present in multiple communities, each of which may have its own value system for research. Even within computer science, ICTD is judged differently by different CS communities.

In the human-computer interaction (HCI) community—for example, at the annual ACM CHI conferences—ICTD has been well received, as HCI

is multidisciplinary by nature and already deals both with quantitative and qualitative research. Moreover, developing-region users differ in their employment and adoption of technology and thus comprise an important research direction for HCI professionals. In fact, HCI is arguably the easiest discipline within CS in which to work on ICTD research. Other areas with some inherent compatibility include systems, networking, databases, and AI. For example, in systems and networking, which are not as multidisciplinary as HCI and more quantitative in character, ICTD work is less natural, but it can still fit well when technology innovation and novel usage are involved. Examples from top-tier conferences include work on delay-tolerant networking, distributed storage, and novel MAC-layer protocols for long-distance WiFi. In these kinds of approaches to ICTD research, there must be a core technical nugget in addition to real-world deployments.

However, research requires a great deal of effort per published report, given the challenges of deployments; over the long term, ICTD researchers must aim to produce papers that are fewer in number but of higher impact. Moreover, it must be noted that ICTD tends to be driven by the solving of a problem rather than by technological innovation (often, in search of a problem), which means that many ICTD projects may not have a core technical nugget after all. Such problems, although highly satisfying to solve, are harder to claim as CS research.

For most projects, the real research is in actually discovering the specification of the problem via repeated field-work and deployments, which is similar in feel to iterative design in HCI. Although HCI is an exception, CS does not generally value problem discovery, especially if the end solution is simple (had we known to apply it). Researcher Matthew Kam went through such iteration to create effective educational games on cellphones:<sup>3</sup>

- ▶ Evaluating 35 existing games for PCs with village students.
- ▶ Creating 10 test games for English as a second language (ESL) and testing them with 47 students.
- ▶ Studying 28 traditional village games to make the games more intui-

tive (compared to Western games).

- ▶ Implementing a new set of games.
- ▶ Leading an ongoing multiyear study on the educational value of these games.

Overall, this process has taken over four years and continues to this day.

ICTD is also developing its own community values over time. The clearest values so far are novelty and on-the-ground empirical results, both quantitative and qualitative. Less clear are the values surrounding repeatability, rigor, and *generalizability*, and least clear is how to merge the values of qualitative fields such as anthropology or ethnography with those of CS. Consider *generalizability*: CS values generalizable results as an indicator of potential impact, while qualitative researchers often emphasize the differences in groups or users and aim to broaden the dialogue. This leads to placing value on reusable technology frameworks, such as HCI toolkits, that can be customized and easily localized. We discuss one such framework here for mixed paper/phone applications. ICTD is also creating its own scholarly forums for discussing and disseminating this work. The International Conference on Information and Communication Technologies and Development and the International Conference on Social Implications of Computers in Developing Countries are two examples.

### What about Sustainability?

Long-term impact requires that ICTD projects be self-sustaining. First, after the researchers leave and the money stops flowing, does the project continue? Second, can it be replicated in other contexts?

Sustainability is challenging to define, and researchers disagree on the details. Most agree on *financial sustainability* as a key element: the deployment must produce enough income to at least cover its costs. In this view, philanthropy is acceptable for “kick starting” a project, but not for supporting routine operational costs. Similarly, while projects typically need not be wildly profitable, they should at least be cash-flow positive, as credit can be challenging.

The operating-cost issues add significant constraints to ICTD solutions.

They include not just the cost of the technology but also availability (uptime), power requirements, potential for theft, and logistics. One common approach to financial sustainability is to commercialize a solution; this has worked well for mobile phones and treadle pumps, for example. Even if a for-profit venture is not the purpose, researchers must essentially address the same issues of costs, cash flow, awareness (marketing), and ongoing support.

*Operational sustainability* is the capacity of the permanent staff to keep the project going technically (without the researchers). In theory, financial sustainability enables operational sustainability (by paying for it), but in practice it cannot do so all by itself. This is because of limits on local skills, supplies, and logistics. Solutions must be not only easy to use, but also amenable to straightforward diagnosis and repair with limited training.

Training costs are actually underrated. ICTD projects, particularly in rural areas, cannot view training as a one-time activity needed only when the project starts. Once trained, IT workers are often tempted to leave for better jobs in urban areas or other countries. Thus training is a recurring cost, and it must be short and effective.

These kinds of sustainability are fundamental to scaling a successful pilot project. Unfortunately, development-work pilots rarely turn into large-scale self-sustaining successes. Typically the pilot is small enough and has enough researchers involved (with their own support) that the financial and operational issues do not really hinder it. Thus the pilot is mostly useful to validate prototypes and assess community reactions. The understanding of financial sustainability requires a longer trial with detailed accounting and no hidden subsidies (unless they are expected to continue at scale); it also requires dealing with replacement costs and expected equipment lifetimes. Operational sustainability must be evaluated via detailed tracking of problems and how and by whom they were solved. In both cases, the system evolves to reduce costs or simplify operation.

Finally, *replication* is the process of moving a successful project to a new environment. As developing regions

are quite heterogeneous in many respects, projects typically need some adjustments to work well with new partners, a different culture, or a different government. Both scaling and replication are active areas of multidisciplinary research, and CS has a critical role to play, given its direct impact on sustainability.

### A Few Sample Projects

We have selected four sample ICTD projects that illustrate some of the issues discussed thus far. The projects focus on four different topics—telemedicine, assistive technology, microfinance, and education—all in the context of developing regions. Each example highlights different challenges and characteristics of the ICTD field. Together, these projects reflect CS-related innovation in the areas of systems, networking, HCI, and AI. The past proceedings of the International Conference on Information and Com-

munication Technologies and Development offer a much larger sampling of current or recent research efforts in the ICTD field. Several other examples and an overview of ICTD are also provided in a recently published special edition of *IEEE Computer*.<sup>8</sup>

**Rural Telemedicine:** The Aravind Eye Care Hospital in southern India is a world leader in high-volume low-cost eye care. Working in the state of Tamil Nadu, Aravind served over 2.4 million patients last year and performed over 280,000 cataract surgeries. More than half the patients receive free or discounted eye care—they are subsidized by paying customers—and the hospitals have been financially self-sustaining for decades.

Despite this success, until recently Aravind had limited reach into rural areas; patient surveys indicated that most patients came from within 20km of a hospital and that only 7% of rural patients had access to any kind of eye

care. After several iterations, it became clear that the solution was to create rural vision centers (VCs) consisting of 1–2 rooms, a nurse, a technician (to make eyeglasses), and notably the means for high-quality doctor/patient videoconferencing. This “video solution,”<sup>7</sup> developed at UC Berkeley, uses novel long-distance WiFi links that are low-cost, low-power, and typically deliver 4Mb/s–6Mb/s between the hospital and the VC over distances ranging from a few to tens of kilometers. (The same basic technology has also been extended to go 382km in Venezuela.)

Having successfully completed a five-VC pilot in early 2006, Aravind now has 24 VCs in operation via a mix of WiFi and DSL (in more urban areas). Some 5,000 patients use the video service per month, with over 100,000 through the end of 2008 having used the WiFi links. Of these 100,000, over 15,000 were effectively blind (primarily due to refractive problems or cataracts), but can now



Researchers from CMU's TechBridgeWorld are working with the Mathru School for the Blind outside Bangalore, India, to enhance the teaching and learning process for writing Braille through the use of a low-cost writing tutor that gives audio feedback to students.

see well; 85% of them have been able to return to income generation. This example shows how the combination of basic needs and large volumes in developing regions enables ICTD research to have great impact. Aravind recently won the \$1M 2008 Gates Foundation Award for Global Health, in large part because of the reach of these vision centers.

**Assistive Technology:** The Mathru School for the Blind is a residential facility that provides free education, clothing, food, and health services to visually impaired children from socially and economically deprived families from remote parts of India. The school is located in the residential area of Yelahanka, a suburb of Bangalore. Teaching Braille, the only means of literacy for the blind, is an important part of the curriculum at Mathru. However, learning to write Braille using the traditional and slate and stylus is not an easy process, for several reasons. First, Braille must be

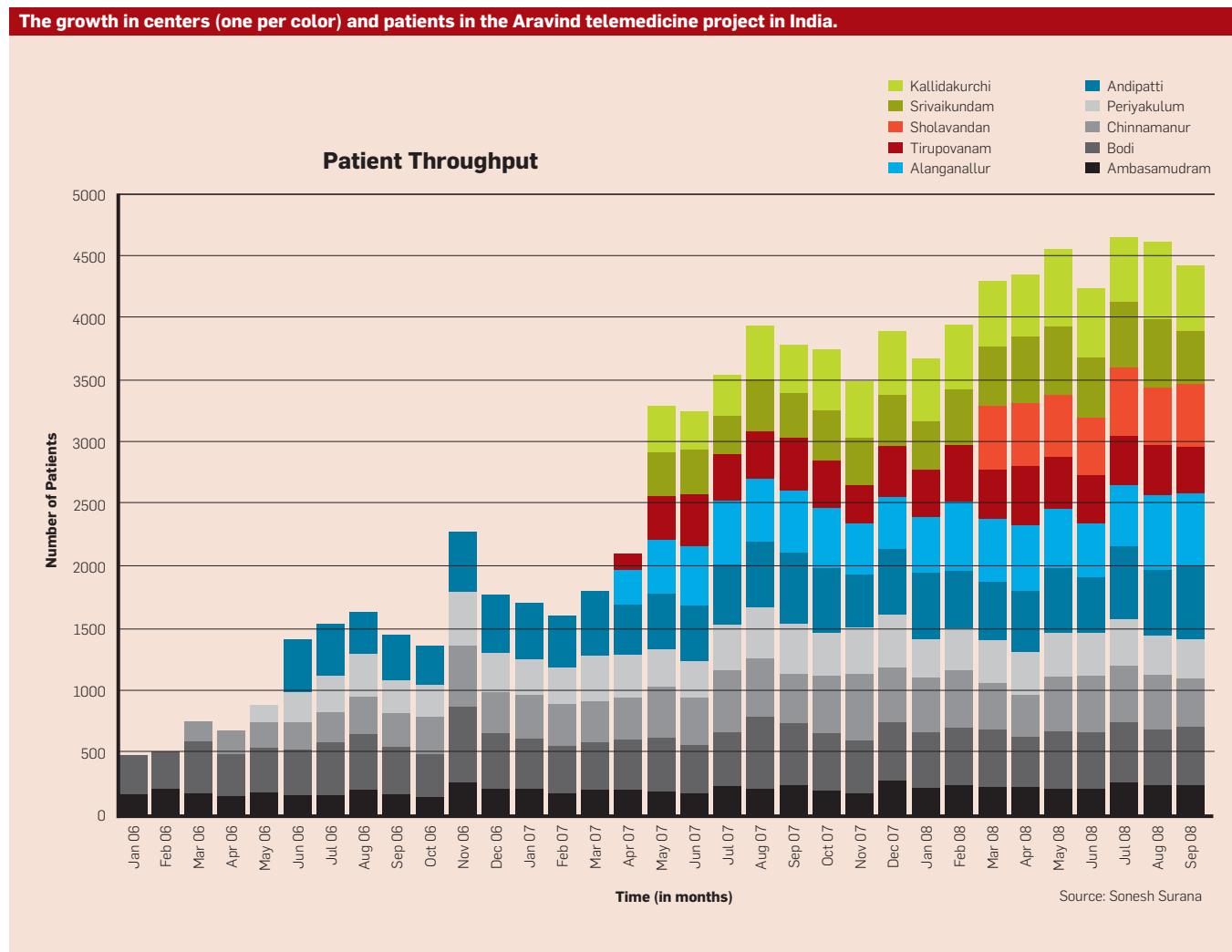
written from right to left in mirror-image format so that the correct Braille characters can be read when the paper is removed from the slate and flipped over. Second, students get delayed feedback; they must wait until their writing is complete and the paper has been removed and read. Third, when the teachers themselves are blind, it is difficult to diagnose problems in the students' writing process by simply reading the end product. Finally, motivation for learning to write Braille is very low because the process is tedious and sometimes even physically taxing for young students.

Researchers from the TechBridge-World group at Carnegie Mellon University are working with Mathru to enhance the teaching and learning process for writing Braille using a slate and stylus. This effort has resulted in a low-cost Braille writing tutor that gives audio feedback to the student as he or she forms characters with the stylus.

This Braille tutor,<sup>2</sup> first designed, implemented, and field-tested in 2006, has been enhanced through an iterative design process to provide several features. They include teaching basic Braille in several languages, teaching basic math symbols, adapting the operational mode to cater to specific student needs, and several educational games that motivate students to learn the skill of writing Braille. This ongoing research has expanded to several new partnerships, including groups in Qatar, Zambia, and China.

**Microfinance Support:** The Nobel Peace Prize for Mohamed Yunus brought overdue attention to the powerful role of microfinance in developing regions. Such services are in dire need of technological support, not only for basic accounting but also to reduce fraud and satisfy government mandates for reporting. The required reports in India, for example, specify multiple copies of the same tables

The growth in centers (one per color) and patients in the Aravind telemedicine project in India.



in different formats, which are easily done with a spreadsheet but tedious and error-prone on paper, which has been the typical mode.

Tapan Parikh, in his dissertation work at University of Washington, developed a system called CAM<sup>6</sup> (short for ‘camera’) that combines the comfort and tangible nature of paper with the power of mobile phones. Two-dimensional barcodes on the paper guide data entry on the phone and help to manage document flow. In addition to workflow support, CAM uses the keypad for numeric input and provides voice feedback, both of which have been well received by semiliterate rural users. This system is now under trial with 400 microfinance groups in India.

*Educational Technology and Technology Education:* Project Kané,<sup>1</sup> an initiative of the TechBridgeWorld group at Carnegie Mellon University, explores the efficacy of technological tools in improving English literacy for children in developing regions, with a focus on Africa. The project started with a three-week pilot study in Ghana that tested the feasibility and impact of using an automated English-reading tutor to improve the level of English literacy among children from low-income families in Accra. This study gave preliminary indications that the tutor had a positive impact on the students’ performance on spelling and fluency tests. It also identified several important factors for success, such as the need to include some local stories familiar to the children and the necessity to narrate the tutorial (on how to use the automated tutor) in a voice with a Ghanaian accent. Based on this initial success, the pilot was scaled to a six-month study that included three groups of children from very different socioeconomic backgrounds, and it has also been replicated in Mongu, Zambia.

The automated tutor used in these studies was not designed for developing regions, however, and it was clear that new educational-technology tools with that focus were needed. This goal is being pursued through a new partnership between TechBridgeWorld researchers and alumni of the course in robotics and artificial intelligence—Ghana’s first—taught at Ashesi University College.

Ayorkor Mills-Tettey, a doctoral

candidate in robotics at Carnegie Mellon University and a native of Ghana, had spearheaded a collaborative project between TechBridgeWorld and Ashesi University College to design and teach that course<sup>5</sup> at Ashesi, a private, accredited, nonsectarian college dedicated to training a new generation of ethical and entrepreneurial leaders in Africa. The collaboration between the two universities led to a summer course designed and taught with careful consideration of the local context, infrastructure, and resources.

Several students who took this course have now graduated and have followed different employment paths; some headed to industry (including a startup company for developing mobile applications) and others to graduate school. Empowered with a strong technology education, some of these students are now collaborating with TechBridgeWorld researchers to design, implement, and field-test educational technology tools to improve literacy in their homeland.

### Looking to the Future

We believe that technology, along with good governance and macroeconomics, represents the path forward for the majority of the world’s people. Consider that in 1970, South Korean and African incomes were similar; but the rapid relative rise of South Korea shows what is possible, due in large part to technology.<sup>4</sup> We believe that proactive research and development of ICTs appropriate for developing regions can lead to similar growth and prosperity over time and to an improved quality of life in the immediate future.

Today we have lots of examples and anecdotes about high impact from ICTD in developing regions, but the field remains ad hoc and largely without the benefit of the innovative thinking that more computer scientists would bring to bear. The situation could change substantially, however. The core costs of computing and communication have dropped to a point that enables CS to affect everyone, especially when combined with the flexibility inherent in software that enables low-cost customization for a wide variety of contexts. This combination makes CS uniquely positioned among all disciplines to have imme-

diate and large-scale impact. But the role of CS in development is essentially a community decision, involving whether we value this work or not. For example, will ICTD be a viable path to a tenure-track CS faculty position?

We can say that although the challenges are great, ICTD is both intellectually rewarding and very attractive to students at all levels. With several recent reports citing the dwindling numbers of students interested in studying CS, perhaps ICTD is one answer. It may help motivate a new generation of computer scientists to contribute their knowledge, talents, and energies toward solving some of the world’s most pressing problems. ■

### References

1. Dias, M.B., Mills-Tettey, G.A., and Mertz, J. The TechBridgeWorld Initiative: Broadening perspectives in computing technology, education, and research. In *Proceedings of the International Symposium on Women and ICTD: Creating Global Transformation*. ACM Press, NY (June 2005).
2. Kalra, N., Lauwers, T., Dewey, D., Stepleton, T., and Dias, M.B. Iterative design of a Braille writing tutor to combat illiteracy. In *Proceedings of the 2nd IEEE/ACM International Conference on Information and Communication Technologies and Development* (Dec. 2007).
3. Kam, M., Ramachandran, D., Devanathan, V., Tewari, A., and Canny, J. Localized iterative design for language learning in underdeveloped regions: The PACE Framework. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (San Jose, CA, Apr. 28–May 3, 2007).
4. Kim, S.J. Information technology and its impact on economic growth and productivity in Korea. *International Economic Journal* 17, 3 (Oct. 2003), 55–75.
5. Mills-Tettey, G.A., Dias, M.B., Browning, B., and Amanquah, N. Teaching technical creativity through robotics: A case study in Ghana. In *Proceedings of the 2nd AI in ICT for Development Workshop, 20th International Joint Conference on Artificial Intelligence* (Jan. 2007).
6. Parikh, T.S., Javid, P., Sasikumar, K., Ghosh, K., and Toyama, K. Mobile phones and paper documents: Evaluating a new approach for capturing microfinance data in rural India. In *Proceedings of the ACM Conference on Computer-Human Interaction* (Apr. 24–27, 2006, Montreal, Canada).
7. Surana, S., Patra, R., Nedevschi, S., and Brewer, E. Deploying a rural wireless telemedicine system: Experiences in sustainability. *IEEE Computer* 41, 6 (June 2008), 48–56.
8. Toyama, K. and Dias, M.B., guest editors. *IEEE Computer Magazine, Special Edition on Information Communication Technology for Development* (June 2008).

**M. Bernadine Dias** ([mbdias@ri.cmu.edu](mailto:mbdias@ri.cmu.edu)) is an assistant research professor at the Robotics Institute of Carnegie Mellon University, Pittsburgh, PA. She founded and directs the TechBridgeWorld group ([www.techbridgeworld.org](http://www.techbridgeworld.org)), which pursues technology research relevant to, and in partnership with, underserved communities throughout the globe.

**Eric Brewer** ([brewer@cs.berkeley.edu](mailto:brewer@cs.berkeley.edu)) is a professor in the computer science division at the University of California, Berkeley. He founded the Federal Search Foundation, which built FirstGov (now USA.gov), the portal for the U.S. government, and he was the founder and chief scientist of the Inktomi Corporation, now part of Yahoo!

# research highlights

---

P. 82

## **Technical Perspective Reframing Security for the Web**

By Andrew Myers

P. 83

## **Securing Frame Communication in Browsers**

By Adam Barth, Collin Jackson, and John C. Mitchell

---

P. 92

## **Technical Perspective Software and Hardware Support for Deterministic Replay of Parallel Programs**

By Norman P. Jouppi

---

P. 93

## **Two Hardware-Based Approaches for Deterministic Multiprocessor Replay**

By Derek R. Hower, Pablo Montesinos, Luis Ceze,  
Mark D. Hill, and Josep Torrellas

---

# Technical Perspective Reframing Security for the Web

By Andrew Myers

THE WEB HAS brought exciting new functionality while simultaneously requiring new mechanisms to make it secure. We've repeatedly discovered that these mechanisms are not good enough, as clever hackers and academics have figured out how to circumvent and misuse them to compromise security.

We now live in a world in which viewing an advertisement might compromise your bank account. In the following paper, "Securing Frame Communication in Browsers," researchers Adam Barth, Collin Jackson, and John Mitchell not only illustrate how subtle some of these security vulnerabilities can be, they show how to solve them in a principled way. This paper has had a real impact: their solutions have already been widely adopted.

Why is Web security difficult? It's because the Web browser is a place where programs and data from different sources interact. Each source may control resources whose security can be affected by the programs and data from other sources. In fact, there is a deep, underlying problem that has never been satisfactorily solved: how to securely permit fine-grained sharing and communication between programs from mutually distrusting sources. Conventionally, security was considered the job of the operating system. But the granularity of operating system enforcement is far too coarse for Web applications, whose security depends on the precise details of the interactions between application-level data structures such as frames, cookies, and interpreted application code.

Web security forces us to think anew about the problem of fine-grained sharing across trust domains because

many exciting new applications and services require this sharing. Some of the techniques developed for operating system security, such as controlled communication between processes, can be adapted to the Web. But Web security poses new challenges as well. For example, Web security violations can occur within the context of a single Web page, which often comprises multiple frames controlled by code from different sources. These frames may be third-party advertisements or integrated content from multiple parties who do not trust each other; the many mashups based on Google Maps are examples of the latter. The absence of effective solutions to the problem of fine-grained interaction between trust domains—coexisting on the very same Web page—has left Web applications vulnerable.

Fortunately, researchers like Barth, Jackson, and Mitchell are applying principled methods to identify and eliminate these vulnerabilities. The vulnerabilities they address arise from

---

**The paper is  
a great example  
of research that  
has impact  
precisely because  
it offers principled  
solutions.**

the feature of frame navigation in Web browsers. Code running in one frame (that is, one trust domain) can control where another frame loads its content from. The authors use elegant reasoning to identify the most permissive secure policy for controlling frame navigation. This argument is so simple and convincing that the policy they identify has been adopted by most major browsers.

In itself, this would be a significant contribution, but the paper goes farther. It newly identifies vulnerabilities in two important mechanisms for communication between different frames; one of these mechanisms is in the HTML 5 standard. The paper gives a thoughtful and principled analysis of each communication mechanism and identifies a fix for each. These fixes have also been adopted by current browsers and communication libraries.

The paper is a great example of research that has impact precisely because it offers principled solutions. Too often, proposed computer security mechanisms merely raise the bar against attacks, starting the next phase of an arms race. This is a different kind of work—work that clearly identifies and convincingly solves a real security problem. The work described in this paper makes our lives more secure and helps the next generation of applications to be built securely. And their work also helps us understand how to think about the new security challenges that lie ahead. C

---

**Andrew Myers** is an associate professor of computer science at Cornell University, Ithaca, NY.

© 2009 ACM 0001-0782/09/0600 \$10.00

# Securing Frame Communication in Browsers

By Adam Barth, Collin Jackson, and John C. Mitchell

## Abstract

**Many Web sites embed third-party content in frames, relying on the browser's security policy to protect against malicious content. However, frames provide insufficient isolation in browsers that let framed content navigate other frames. We evaluate existing frame navigation policies and advocate a stricter policy, which we deploy in the open-source browsers. In addition to preventing undesirable interactions, the browser's strict isolation policy also affects communication between cooperating frames. We therefore analyze two techniques for interframe communication between isolated frames. The first method, fragment identifier messaging, initially provides confidentiality without authentication, which we repair using concepts from a well-known network protocol. The second method, postMessage, initially provides authentication, but we discover an attack that breaches confidentiality. We propose improvements in the postMessage API to provide confidentiality; our proposal has been standardized and adopted in browser implementations.**

## 1. INTRODUCTION

Web sites contain content from sources of varying trustworthiness. For example, many Web sites contain third-party advertising supplied by advertisement networks or their sub-syndicates.<sup>3</sup> Other common aggregations of third-party content include Flickr albums, Facebook badges, and personalized home pages offered by the three major Web portals (iGoogle, My Yahoo! and Windows Live). More advanced uses of third-party components include Yelp's use of Google Maps to display restaurant locations, and the Windows Live Contacts gadget. A Web site combining content from multiple sources is called a *mashup*, with the party combining the content called the *integrator*, and integrated content called a *gadget*. In simple mashups, the integrator does not intend to communicate with the gadgets and requires only that the browser provide isolation. In more sophisticated mashups, the integrator does wish to communicate and requires secure interframe communication. When a site wishes to provide isolation and communication between content on its pages, the site inevitably relies on the browser rendering process and isolation policy, because Web content is rendered and viewed under browser control.

In this paper, we study a contemporary Web version of a recurring problem in computer systems: isolating untrusted, or partially trusted, components while providing secure intercomponent communication. Whenever a site integrates third-party content, such as an advertisement, a

map, or a photo album, the site runs the risk of incorporating malicious content. Without isolation, malicious content can compromise the confidentiality and integrity of the user's session with the integrator. Although the browser's well-known "same-origin policy"<sup>19</sup> restricts script running in one frame from manipulating content in another frame, browsers use a different policy to determine whether one frame is allowed to navigate (change the location of) another. Although browsers must restrict navigation to provide isolation, navigation is the basis of one form of interframe communication used by leading companies and navigation can be used to attack a second interframe communication mechanism.

Many recent browsers have overly permissive frame navigation policies that lead to a variety of attacks. To prevent attacks, we demonstrate against the Google AdSense login page and the iGoogle gadget aggregator, we propose tightening the browser's frame navigation policy. Based on a comparison of four policies, we advocate a specific policy that restricts navigation while maintaining compatibility with existing Web content. We have collaborated with the HTML 5 working group to standardize this policy and with browser vendors to deploy this policy in Firefox 3, Safari 3.1, and Google Chrome. Because the policy is already implemented in Internet Explorer 7, our preferred policy is now standardized and deployed in the four most-used browsers.

With strong isolation, frames are limited in their interactions, raising the issue of how isolated frames can cooperate as part of a mashup. We analyze two techniques for interframe communication: fragment identifier messaging and postMessage. Table 1 summarizes our results.

- Fragment identifier messaging uses frame navigation to send messages between frames. This channel lacks an important security property: messages are confidential but senders are not authenticated. These properties are analogous to a network channel in which senders encrypt their messages with the recipient's public key. The Microsoft.Live.Channels library uses fragment identifier messaging to let the Windows Live Contacts gadget communicate with its integrator, following an authentication protocol analogous to the Needham-Schroeder public-key protocol.<sup>17</sup>

The original version of this paper was published in the *Proceedings of the 17th USENIX Security Symposium*, July 2008.

**Table 1: Security properties of frame communication channels.**

	<b>Confidentiality</b>	<b>Authentication</b>	<b>Network Analogue</b>
Fragment identifier messaging	✓		Public Key Encryption
Original postMessage		✓	Public Key Signatures
Improved postMessage	✓	✓	SSL/TLS

We discover an attack on this protocol, related to Lowe's anomaly in the Needham-Schroeder protocol,<sup>15</sup> in which a malicious gadget can impersonate the integrator to the Contacts gadget. We suggested a solution based on Lowe's improvement to the Needham-Schroeder protocol<sup>15</sup> that Microsoft implemented and deployed.

- `postMessage` is a browser API designed for interframe communication<sup>10</sup> that is implemented in Internet Explorer 8, Firefox 3, Safari 4, Google Chrome, and Opera. Although `postMessage` has been deployed in Opera since 2005, we demonstrate an attack on the channel's confidentiality using frame navigation. In light of this attack, the `postMessage` channel provides authentication but lacks confidentiality, analogous to a channel in which senders cryptographically sign their messages. To secure the channel, we propose modifying the API. Our proposal has been adopted by the HTML 5 working group and all the major browsers.

The remainder of the paper is organized as follows. Section 2 details our threat models. Section 3 surveys existing frame navigation policies and standardizes a secure policy. Section 4 analyzes two frame communication mechanisms, demonstrates attacks, and proposes defenses. Section 5 describes related work. Section 6 concludes.

## 2. THREAT MODEL

In this section, we define precise threat models so that we can determine how effectively browser mechanisms defend against specific classes of attacks. We consider two kinds of attackers, a "Web attacker" and a slightly more powerful "gadget attacker." Although *phishing*<sup>4,6</sup> can be described informally as a Web attack, we do not assume that either the Web attacker or the gadget attacker can fool the user by using a confusing domain name (such as `bankofthevest.com`) or by other social engineering. Instead, we assume the user uses every browser security feature, including the location bar and lock icon, accurately and correctly.

### 2.1. Web attacker

A *Web attacker* is a malicious principal who owns one or more machines on the network. To study the browser security policy, we assume that the user's browser renders content from the attacker's Web site.

- **Network Abilities:** The Web attacker has no special network abilities. In particular, the Web attacker can send and receive network messages only from machines

under his or her control, possibly acting as a client or server in network protocols of the attacker's choice. Typically, the Web attacker uses at least one machine as an HTTP server, which we refer to as `attacker.com`. The Web attacker has HTTPS certificates for domains he or she owns; certificate authorities provide such certificates for free. The Web attacker's network abilities are decidedly *weaker* than the usual network attacker considered in network security because the Web attacker can neither eavesdrop on messages to nor forge messages from other network locations. For example, a Web attacker cannot be a network "man-in-the-middle."

- **Client Abilities:** We assume that the user views `attacker.com` in a popular browser, rendering the attacker's content. We make this assumption because an honest user's interaction with an honest site should be secure even if the user visits a malicious site in another browser window. The Web attacker's content is subject to the browser's security policy, making the Web attacker decidedly *weaker* than an attacker who can execute an arbitrary code with the user's privileges. For example, a Web attacker cannot install a system-wide key logger or botnet client.

We do not assume that the user treats `attacker.com` as a site other than `attacker.com`. For example, the user never gives a `bank.com` password to `attacker.com`. We also assume that honest sites are free of cross-site scripting vulnerabilities.<sup>20</sup> In fact, none of the attacks described in this paper rely on running malicious JavaScript as an honest principal. Instead, we focus on privileges the browser itself affords the attacker to interact with honest sites.

In addition to our interest in protecting users that visit malicious sites, our assumption that the user visits `attacker.com` is further supported by several techniques for attracting users. For example, an attacker can place Web advertisements, host popular content with organic appeal, or send bulk e-mail encouraging visitors. Typically, simply viewing an attacker's advertisement (such as on a search page) lets the attacker mount a Web attack. In a previous study,<sup>12</sup> we purchased over 50,000 impressions for \$30. During each of these impressions, a user's browser rendered our content, giving us the access required to mount a Web attack.

Attacks accessible to a Web attacker have significant practical impact because these attacks do not require unusual control of the network. Web attacks can also be carried out by a standard man-in-the-middle network attacker, once the user visits a single HTTP site, because a man-in-the-middle

can inject malicious content into the HTTP response, simulating a reply from `attacker.com`.

## 2.2. Gadget attacker

A *gadget attacker* is a Web attacker with one additional ability: the integrator embeds a gadget of the attacker's choice. This assumption lets us accurately evaluate mashup isolation and communication protocols because the purpose of these protocols is to let an integrator embed untrusted gadgets safely. In practice, a gadget attacker can either wait for the user to visit the integrator or can redirect the user to the integrator's Web site from `attacker.com`.

## 3. FRAME ISOLATION

Web sites can use frames to delegate portions of their screen real estate to other Web sites. For example, a site can sell parts of their pages to advertising networks. The browser displays the location of the main, or *top-level*, frame in its location bar. Subframes are often visually indistinguishable from other parts of a page, and the browser does not display their location in its user interface.

### 3.1. Background

The browser's scripting policy answers the question "when can one frame manipulate the contents of another frame?" The scripting policy is the most important part of the browser security policy because a frame can act on behalf of every other frame it can script. For example,

```
otherWindow.document.forms[0].password.value
```

attempts to read the user's password from another window. Modern Web browsers let one frame read and write all the properties of another frame only when their content was retrieved from the same *origin*, i.e. when the scheme (e.g., `http` or `https`), host, and port of their locations match. If the content of `otherWindow` was retrieved from a different origin, the browser's security policy will prevent the script above from accessing `otherWindow.document`.

In addition to enforcing the scripting policy, every browser must answer the question "when is one frame permitted to navigate another frame?" Prior to 1999, all Web browsers implemented a permissive policy:

#### Permissive Policy

A frame can navigate any other frame.

For example, if `otherWindow` includes a frame,

```
otherWindow.frames[0].location =
  "https://attacker.com/";
```

navigates the frame to `https://attacker.com/`. Under the permissive policy, the browser navigates `otherWindow`

even if it contains content from another origin. There are a number of idioms for navigating frames, including

```
window.open("https://attacker.com/", "frameName");
```

which navigates a frame named `frameName`. Frame names exist in a global name space that is shared across origins.

### 3.2. Cross-window attacks

In 1999, Georgi Guninski discovered that the permissive frame navigation policy admits serious attacks.<sup>7</sup> At the time, the password field on the CitiBank login page was contained within a frame, and the Web attacker could navigate that frame to `https://attacker.com/`, letting the attacker fill the frame with identical-looking content that steals the password. This *cross-window attack* proceeds as follows:

1. The user views a blog that displays the attacker's ad.
2. Separately, the user visits `bank.com`, which displays its password field in a frame.
3. The advertisement navigates the password frame to `https://attacker.com/`. The location bar remains `https://bank.com` and the lock icon remains present.
4. The user enters his or her `bank.com` password into the `https://attacker.com/` frame on the `bank.com` page, submitting the password to `attacker.com`.

Of the browsers in heavy use today, Internet Explorer 6 and Safari 3 both implement the permissive policy and allow this attack. Internet Explorer 7 and Firefox 2 implement stricter policies (described in subsequent sections). Many Web sites, including Google AdSense, display their password field in a frame and are vulnerable to this attack; see Figure 1.

### 3.3. Same-window attacks

In 2001, Mozilla prevented the cross-window attack by implementing a stricter policy:

#### Window Policy

A frame can navigate only frames in its window.

**Figure 1: Cross-window attack. The attacker hijacks the password field, which is in a frame.**



This policy prevents the cross-window attack because the Web attacker does not control a frame in a trusted window and, without a foothold in the window, the attacker cannot navigate the login frame. However, the window policy is insufficiently strict to protect users because the gadget attacker does have a foothold in a trusted window in a mashup. (Recall that, in a mashup, the integrator combines gadgets from different sources into a single experience.)

- **Aggregators:** Gadget aggregators, such as iGoogle, My Yahoo! and Windows Live, provide one form of mashup. These sites let users customize their experience by including gadgets (such as stock tickers, weather predictions, and news feeds) on their home page. These sites put third-party gadgets in frames and rely on the browser to protect users from malicious gadgets.
- **Advertisements:** Web advertising produces mashups that combine first-party content, such as news articles or sports statistics, with third-party advertisements. Most advertisements, including Google AdWords, are contained in frames, both to prevent the advertisers (who provide the gadgets) from interfering with the publisher's site and to prevent the publisher from using JavaScript to click on the advertisements.

We refer to pages with advertisements as *simple mashups* because the integrator and the gadgets do not communicate. Simple mashups rely on the browser to provide isolation but do not require interframe communication.

The windows policy offers no protection for mashups because the integrator's window contains untrusted gadgets. A gadget attacker who supplies a malicious gadget does control a frame in the honest integrator's window, giving the attacker the foothold required to mount a *gadget hijacking* attack.<sup>14</sup> A malicious gadget can navigate a target gadget to `attacker.com` and impersonates the gadget to the user. For example, iGoogle is vulnerable to gadget hijacking in browsers, such as Firefox 2, that implement the permissive or window policies; see Figure 2. Consider an iGoogle gadget that lets users access their Hotmail account. If the user is not logged into Hotmail, the gadget requests the user's Hotmail password. A malicious gadget can replace the Hotmail gadget with and steal the user's Hotmail password. As in the cross-window attack, the user is unable to distinguish the malicious password field from the honest password field.

### 3.4. Stricter policies

Although browser vendors do not document their navigation policies, we reverse engineered the policies of existing browsers (see Table 2). In addition to the permissive and window policies, we found two other policies:

#### *Descendant Policy*

A frame can navigate only its descendants.

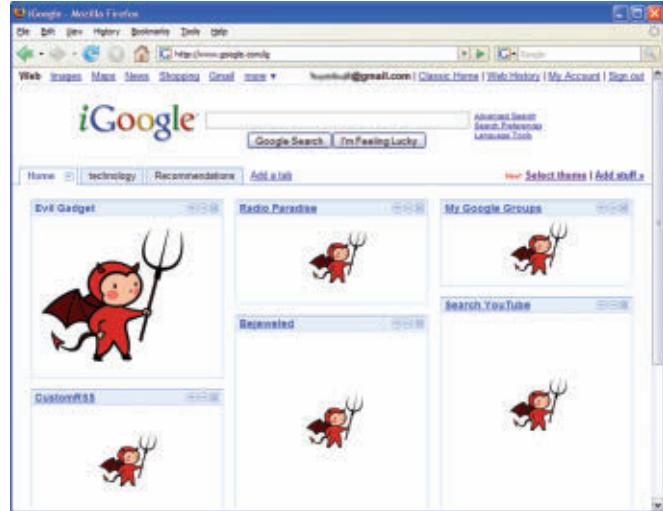
#### *Child Policy*

A frame can navigate only its direct children.

**Figure 2: Gadget hijacking.** Under the window policy, the attacker gadget can navigate other gadgets.



(a) Before



(b) After

The Internet Explorer 6 team wanted to enable the child policy by default but shipped the permissive policy because the child policy was incompatible with a large number of Web sites. The Internet Explorer 7 team designed the descendant policy to balance the security requirement to defeat the cross-window attack with the compatibility requirement to support existing sites.<sup>18</sup>

To select a frame navigation policy that provides the best trade-off between security and compatibility, we appeal to the principle of *pixel delegation*. When one frame embeds a child frame, the parent frame delegates a region of the screen to the child frame. The browser prevents the child frame from drawing outside of its bounding box but does allow the parent frame to draw over the child using the `position: absolute` style. Frame navigation attacks hinge on the attacker escalating his or her privileges and drawing on otherwise inaccessible regions of the screen. The descendant policy is the most permissive (and therefore

**Table 2: Frame navigation policies deployed in existing browsers prior to our work.**

IE 6 (Default)	IE 6 (Optional)	IE 7 (Default)	IE 7 (Optional)	Firefox 2	Safari 3	Opera 9
Permissive	Child	Descendant	Permissive	Window	Permissive	Child

most compatible) policy that prevents the attacker from overwriting screen real estate “belonging” to another origin. Although the child policy is stricter than the descendant policy, the added strictness does not provide a significant security benefit because the attacker can simulate the visual effects of navigating a grandchild frame by drawing over the region of the screen occupied by the grandchild frame. The child policy’s added strictness does, however, reduce the policy’s compatibility with existing sites, discouraging browser vendors from deploying the child policy.

Maximizing the compatibility of the descendant policy requires taking the browser’s scripting policy into account. Consider one site that embeds two child frames from a second origin. Should one of those child frames be permitted to navigate its sibling? Strictly construed, the descendant policy forbids this navigation because the target frame is a sibling, not a descendant. However, this navigation should be allowed because an attacker can perform the navigation by injecting a script into the sibling frame that causes the frame to navigate itself. The browser lets the attacker inject this script because the two frames are from the same origin. More generally, the browser can maximize the compatibility of the descendant policy by recognizing *origin propagation* and letting an active frame navigate a target frame if the target frame is the descendant of a frame in the same origin as the active frame. Defined in this way, the frame navigation policy avoids creating a suborigin privilege.<sup>11</sup> This added permissiveness does not sacrifice security because an attacker can perform the same navigations indirectly, but the refined policy is more convenient for honest Web developers.

We collaborated with the HTML 5 working group<sup>9</sup> and standardized the descendant policy in the HTML 5 specification. The descendant policy has now been adopted by Internet Explorer 7, Firefox 3, Safari 3.1, and Google Chrome. We also reported a vulnerability in Flash Player that could be used to bypass Internet Explorer 7’s frame navigation policy. Adobe fixed this vulnerability in a security update.

#### 4. FRAME COMMUNICATION

Unlike simple aggregators and advertisements, sophisticated mashups comprise gadgets that communicate with each other and with their integrator. For example, Yelp integrates the Google Maps gadget, illustrating the need for secure interframe communication in real deployments. Google provides two versions of its Maps gadget:

- **Frame:** In the frame version, the integrator embeds a frame to maps.google.com, in which Google displays a map of the specified location. The user can interact with the map, but the integrator cannot.
- **Script:** In the script version, the integrator embeds a <script> tag that runs JavaScript from maps.

google.com. This script creates a rich JavaScript API that the integrator can use to interact with the map, but the script runs with all of the integrator’s privileges.

Yelp, a popular review Web site, uses the Google Maps gadget to display the locations of restaurants and other businesses. Yelp requires a high degree of interactivity with the Maps gadget because it places markers on the map for each restaurant and displays the restaurant’s review when the user clicks on the marker. To deliver these advanced features, Yelp must use the script version of the Maps gadget, but this design requires Yelp to trust Google Maps completely because Google’s script runs with Yelp’s privileges, granting Google the ability to manipulate Yelp’s reviews and steal Yelp’s customer’s information. Although Google might be trustworthy, the script approach does not scale beyond highly respected gadget providers. Secure interframe communication promises the best of both alternatives: sites with functionality like Yelp can realize the interactivity of the script version of Google Maps gadget while maintaining the security of the frame version of the gadget.

#### 4.1. Fragment identifier messaging

Although the browser’s scripting policy isolates frames from different origins, clever mashup designers have discovered an unintended channel between frames, *fragment identifier messaging*,<sup>1, 21</sup> which is regulated by the browser’s less-restrictive frame navigation policy. This “found” technology lets mashup developers place each gadget in a separate frame and rely on the browser’s security policy to prevent malicious gadgets from attacking the integrator and honest gadgets. We analyze fragment identifier messaging in use prior to our analysis and propose improvements that have since been adopted.

**Mechanism:** Normally, when a frame is navigated to a new URL, the browser requests the URL from the network and replaces the frame’s document with the retrieved content. However, if the new URL matches the old URL everywhere except in the fragment (the part after the #), then the browser does not reload the frame. If frames[0] is currently located at http://example.com/doc,

```
frames[0].location = "http://example.com/doc#msg";
```

changes the frame’s location without reloading the frame or destroying its JavaScript context. The frame can read its fragment by polling window.location.hash to see if the fragment has changed. This technique can be used to send messages between frames while avoiding network latency.

**Security Properties:** The fragment identifier channel has less-than-ideal security properties. The browser’s scripting

policy prevents other origins from eavesdropping on messages because they are unable to *read* the frame's location (even though the navigation policy lets them *write* the frame's location). Browsers also prevent arbitrary origins from tampering with portions of messages. Other security origins can, however, overwrite the fragment identifier in its entirety, leaving the recipient to guess the sender of each message.

To understand these security properties, we draw an analogy with the well-known properties of network channels. We view the browser as guaranteeing that the fragment identifier channel has *confidentiality*: a message can be read only by its intended recipient. The fragment identifier channel fails to be a secure channel, however, because it lacks *authentication*: a recipient cannot determine the sender of a message unambiguously. The attacker might be able to replay previous messages using the browser's history API.

The fragment identifier channel is analogous to a channel on an untrusted network in which each message is encrypted with the public key of its intended recipient. In both cases, when Alice sends a message to Bob, no one except Bob learns the contents of the message (unless Bob forwards the message). In both settings, the channel does not provide a reliable procedure for determining who sent a given message. There are two key differences between the fragment identifier channel and the public-key channel:

1. Public-key channel is susceptible to traffic analysis, but an attacker cannot determine the length of a message sent over the fragment identifier channel. An attacker can extract timing information by polling the browser's clock, but obtaining high-resolution timing information degrades performance.
2. Fragment identifier channel is constrained by the browser's frame navigation policy. In principle, this could be used to construct protocols secure for the fragment identifier channel that are insecure for the public-key channel (by preventing the attacker from navigating the recipient), but in practice this restriction has not prevented us from constructing attacks on existing implementations.

Despite these differences, we find the network analogy useful in analyzing interframe communication.

**Windows Live Channels:** Microsoft uses fragment identifier messaging in its Windows Live platform library to implement a higher-level channel API, `Microsoft.Live.Channels`.<sup>21</sup> The Windows Live Contacts gadget uses this API to communicate with its integrator. The integrator can instruct the gadget to add or remove contacts from the user's contacts list, and the gadget can send the integrator details about the user's contacts. Whenever the integrator asks the gadget to perform a sensitive action, the gadget asks the user to confirm the operation and displays the integrator's host name to aid the user in making trust decisions. Prior to our analysis, `Microsoft.Live.Channels` used a protocol to add authentication to the fragment identifier channel. By reverse engineering the implementation, we determined

that the library used the following protocol to establish a secure channel:

$$\begin{aligned} A \rightarrow B : N_A, \text{URI}_A \\ B \rightarrow A : N_A, N_B \\ A \rightarrow B : N_B, \text{Message}_1 \end{aligned}$$

In this notation,  $A$  and  $B$  are frames,  $N_A$  and  $N_B$  are fresh nonces (numbers chosen at random during each run of the protocol), and  $\text{URI}_A$  is the location of  $A$ 's frame. Under the network analogy described above, this protocol is analogous to the classic Needham–Schroeder public-key protocol.<sup>17</sup> The Needham–Schroeder protocol was designed to establish a shared secret between two parties over an insecure channel. Instead of using encryption as in the Needham–Schroeder protocol, Windows Live relies on the fragment identifier channel to provide confidentiality.

The Needham–Schroeder public-key protocol has a well-known anomaly, due to Lowe,<sup>15</sup> that leads to an attack in the browser setting. In the Lowe scenario, an honest principal, Alice, initiates the protocol with a dishonest party, Eve. Eve then convinces honest Bob that she is Alice. In order to exploit the Lowe anomaly, an honest principal must be willing to initiate the protocol with a dishonest principal. This requirement is met in mashups because the integrator initiates the protocol with the gadget attacker's gadget when the mashup is initialized. The Lowe anomaly can be exploited to impersonate the integrator to the gadget:

$$\begin{aligned} \text{Integrator} \rightarrow \text{Attacker} : N_I, \text{URI}_I \\ \text{Attacker} \rightarrow \text{Gadget} : N_I, \text{URI}_I \\ \text{Gadget} \rightarrow \text{Integrator} : N_I, N_G \\ \text{Integrator} \rightarrow \text{Attacker} : N_G, \text{Message}_1 \end{aligned}$$

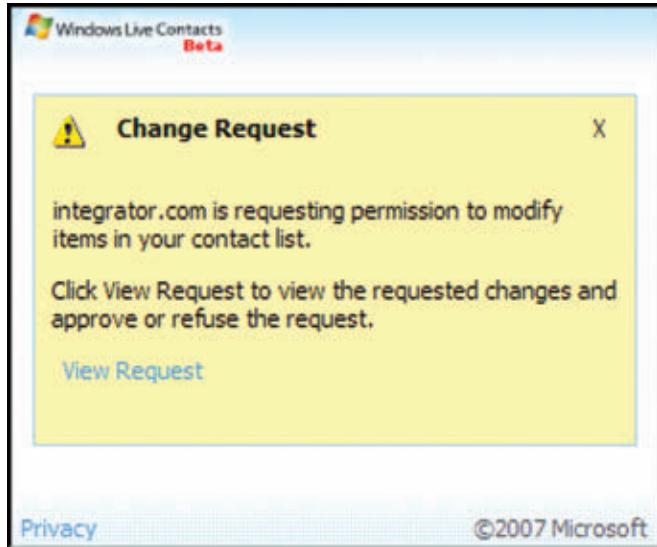
After these four messages, the attacker possesses  $N_I$  and  $N_G$  and can impersonate the integrator to the gadget. We have implemented this attack against the Windows Live Contacts gadget. The anomaly is especially problematic for the Contacts gadget because it displays the integrator's host name to the user in its security user interface (see Figure 3).

**Securing Fragment Identifier Messaging:** The channel can be secured using a variant of the Needham–Schroeder–Lowe protocol.<sup>15</sup> As in Lowe's improvement to the original protocol, we recommend including the responder's identity in the second message of the protocol, letting the honest initiator detect the attack and abort the protocol:

$$\begin{aligned} A \rightarrow B : N_A, \text{URI}_A \\ B \rightarrow A : N_A, N_B, \text{URI}_B \\ A \rightarrow B : N_B \end{aligned}$$

We contacted Microsoft, the OpenAJAX Alliance, and IBM about the vulnerabilities in their fragment identifier messaging protocols. Microsoft and the OpenAJAX Alliance have adopted our suggestions and deployed the above protocol in

**Figure 3: Lowe Anomaly.** The gadget believes the request came from integrator.com, but in reality the request was made by attacker.com.



updated versions of their libraries. IBM adopted our suggestions and revised their SMash<sup>14</sup> paper.

## 4.2. postMessage

HTML 5<sup>10</sup> specifies a new browser API for asynchronous communication between frames. Unlike fragment identifier messaging, postMessage was designed for cross-origin communication. The postMessage API was originally implemented in Opera 8 and is now supported by Internet Explorer 8, Firefox 3, Safari 3.1, and Google Chrome. We discovered a vulnerability in an early version of the API, which has since been eliminated by modifications we suggested. To send a message to another frame, the sender calls the postMessage method:

```
frames[0].postMessage("Hello world.");
```

In the recipient's frame, the browser generates a message event with the message, the origin (scheme, host, and port) of the sender, and a reference to the sender's frame.

**Security Properties:** The postMessage channel guarantees authentication, messages accurately identify their senders, but the channel lacks confidentiality. Thus, postMessage has almost the “opposite” security properties as fragment identifier messaging. The postMessage channel is analogous to a channel on an untrusted network in which each message is cryptographically signed by its sender. In both settings, if Alice sends a message to Bob, Bob can determine unambiguously that Alice sent the message. With postMessage, the origin property identifies the sender; with cryptographic signatures, the signature identifies signer. One difference between the channels is that cryptographic signatures can be easily replayed, but postMessage resists replay attacks.

**Attacks:** We discover an attack that breaches the confidentiality of the postMessage channel. Because a message sent with postMessage is directed at a frame, an attacker can intercept the message by navigating the frame to attacker.com before the browser generates the message event:

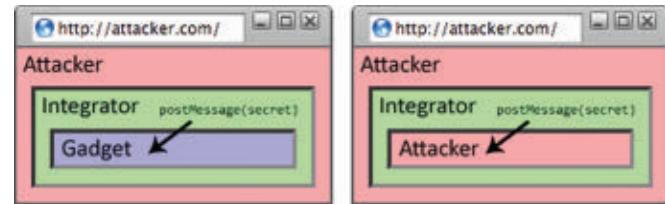
- **Recursive Mashup Attack:** If an integrator calls postMessage on a gadget contained in a frame, the attacker can load the integrator inside a frame and intercept the message by navigating the gadget frame (a descendant of the attacker's frame) to attacker.com. When the integrator calls postMessage on the “gadget's” frame, the browser delivers the message to the attacker (see Figure 4).
- **Reply Attack:** Suppose the integrator uses the origin to decide whether to reply to a message event:

```
if (evt.origin == "https://gadget.com")
    evt.source.postMessage(secret);
```

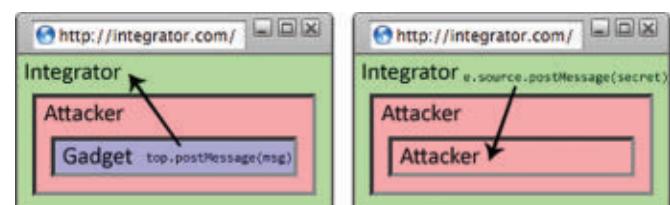
- The attacker can intercept the secret by navigating the source frame before the browser generates the message event. This attack can succeed even under the child frame navigation policy if the honest gadget sends its messages via top.postMessage(...). The attacker's gadget can embed a frame to the honest gadget and navigate the honest gadget before the integrator replies to the “gadget's” frame (see Figure 5).

**Securing postMessage:** Although sites might be able to build a secure channel using the original postMessage API, we recommend that postMessage provide confidentiality natively. In MashupOS,<sup>22</sup> we previously proposed that inter-frame communication APIs should let the sender specify the origin of the intended recipient. Similarly, we propose

**Figure 4: Recursive mashup attack.** The attacker navigates the gadget's frame to attacker.com.



**Figure 5: Reply attack.** The attacker intercepts the integrator's response to the gadget's message.



extending the `postMessage` API with a second parameter: `targetOrigin`. The browser will deliver the message *only if* the frame's current origin matches the specified `targetOrigin`. If the sender uses "\*" as the `targetOrigin`, the browser will deliver the message to any origin. Using this improved API, a frame can reply to a message using the following idiom:

```
if (evt.origin == "https://gadget.com")
    evt.source.postMessage(secret, evt.origin);
```

We implemented this API change as patches for Firefox and Safari. Our proposal was accepted by the HTML 5 working group.<sup>8</sup> The improved API is now available in Internet Explorer 8, Firefox 3, Safari 4, and Google Chrome.

## 5. RELATED WORK

### 5.1. Mitigations for gadget hijacking

SMash<sup>14</sup> mitigates gadget hijacking (also known as “frame phishing”) by carefully monitoring the frame hierarchy and browser events for unexpected navigations. Although neither the integrator nor the gadgets can prevent these navigations, the mashup can alert the user and refuse to function if it detects an illicit navigation. SMash waits 20s for a gadget to load before assuming that the gadget has been hijacked. An attacker might be able to fool the user into entering sensitive information during this interval, but using a shorter interval might cause users with slow network connections to receive spurious warnings. The descendant policy makes such mitigation unnecessary.

### 5.2. Safe subsets of HTML and JavaScript

One way to sidestep the security issues of frame-based mashups is to avoid using frames by combining the gadgets and the integrator into a single document. This approach forgoes the protections afforded by the browser's security policy and requires gadgets to be written in a “safe subset” of HTML and JavaScript that prevents a malicious gadget from attacking the integrator or other gadgets. Several open-source implementations (FBML, ADsafe, and Caja) are available. FBML is currently the most successful subsets and is used by the Facebook Platform.

### 5.3. Subspace

In Subspace,<sup>13</sup> we used a multilevel hierarchy of frames that coordinated their `document.domain` property to communicate directly in JavaScript. Similar to most frame-based mashups, the descendant frame navigation policy is required to prevent gadget hijacking.

### 5.4. Module tag

The proposed `<module>` tag<sup>2</sup> is similar to the `<iframe>` tag, but the module runs in an unprivileged security context, without a principal, and the browser prevents the integrator from overlaying content on top of the module. Unlike `postMessage`, the communication primitive used with the `<module>` tag is explicitly unauthenticated because the module lacks a principal.

### 5.5. Security = restricted and jail

Internet Explorer supports a `security` attribute<sup>16</sup> for frames. When set to `restricted`, the frame's content cannot run JavaScript. Similarly, the proposed `<jail>` tag<sup>5</sup> encloses untrusted content and prevents the jailed content from running JavaScript. Unfortunately, eliminating JavaScript prevents gadgets from offering interactive experiences.

### 5.6. MashupOS

In MashupOS,<sup>22</sup> we proposed new primitives both for isolation and communication. Our improvements to frame navigation policies and `postMessage` let developers realize some of the benefits of MashupOS using existing browsers.

## 6. CONCLUSION

Web sites that combine content from multiple sources can leverage browser frame isolation and interframe communication. Although the browser's same-origin security policy restricts direct access between frames, recent browsers have used differing policies to regulate when one frame may navigate another. The original permissive frame navigation policy admits a number of attacks, and the subsequent window navigation policy leaves mashups vulnerable to similar attacks. The better descendant policy, which we collaborated with the HTML 5 working group to standardize, balances security and compatibility and has been adopted by Internet Explorer 7 (independently), Firefox 3, Safari 3.1, and Google Chrome.

In existing browsers, frame navigation can be used for inter-frame communication via a technique known as fragment identifier messaging. If used directly, fragment identifier messaging lacks authentication. We showed that the authentication protocols used by Windows.Live.Channels, SMash, and OpenAjax 1.1 were vulnerable to attacks but can be repaired in a manner analogous to Lowe's variation of the Needham-Schroeder protocol.<sup>15</sup> This improvement has been adopted by Microsoft Windows Live, IBM Smash,<sup>14</sup> and the OpenAjax Alliance.

Originally, `postMessage`, another interframe communication channel, suffered the converse vulnerability: using frame navigation, an attacker could breach confidentiality. We propose extending the `postMessage` API to provide confidentiality by letting the sender specify an intended recipient. Our proposal has been adopted by the HTML 5 working group, Internet Explorer 8, Firefox 3, Safari 4, Google Chrome, and Opera.

With these improvements, frames provide stronger isolation and better communication, becoming a more attractive feature for integrating third-party Web content. One important area of future work is improving the usability of the browser's security user interface. For example, a gadget is permitted to navigate the top-level frame, redirecting the user from the mashup to a site of the attacker's choice. Although the browser's location bar makes this navigation evident, many users ignore the location bar. Another area for future work is improving isolation in the face of browser implementation errors, which could let a gadget subvert the browser's security mechanisms.

## Acknowledgments

We thank Mike Belzner, Sumeer Bhola, Dan Boneh, Gabriel E. Corvera, Ian Hickson, Koji Kato, Eric Lawrence, Erick Lee, David Lenoe, David Ross, Maciej Stachowiak, Hallvard Steen, Peleus Uhley, Jeff Walden, Sam Weinig, and Boris Zbarsky for their helpful suggestions and feedback. This work is supported by grants from the National Science Foundation and the US Department of Homeland Security. C

## References

1. Burke, J. Cross domain frame communication with fragment identifiers. <http://tagneto.blogspot.com/2006/06/cross-domain-frame-communication-with.html>.
2. Crockford, D. The <module> tag. <http://www.json.org/module.html>.
3. Daswani, N., Stoppelman, M. et al. The anatomy of Clickbot.A. In Proceedings of the HotBots (2007).
4. Dhamija, R., Tygar, J.D., Hearst, M. Why phishing works. In CHI '06: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2006).
5. Eich, B. JavaScript: Mobility and ubiquity. [http://kathrin.dagstuhl.de/files/Materials/07/07091/07091\\_EichBrendan.Slides.pdf](http://kathrin.dagstuhl.de/files/Materials/07/07091/07091_EichBrendan.Slides.pdf).
6. Felten, E.W., Balfanz, D., Dean, D., Wallach, D.S. Web spoofing: An Internet con game. In Proceedings of the 20th National Information Systems Security Conference (1996).
7. Guninski, G. Frame spoofing using loading two frames. Mozilla Bug 13871.
8. Hickson, I. Re: A potential slight security enhancement to postMessage, February 2008. <http://lists.whatwg.org/pipermail/whatwg-whatwg.org/2008-February/013949.html>.
9. Hickson, I. Re: HTML5 frame navigation policy, April 2008. <http://lists.whatwg.org/pipermail/whatwg-whatwg.org/2008-April/014597.html>.
10. Hickson, I. et al. HTML 5 Working Draft. <http://www.whatwg.org/specs/web-apps/current-work/>.
11. Jackson, C., Barth, A. Beware of finer-grained origins. In Proceedings of the Web 2.0 Security and Privacy (W2SP) (2008).
12. Jackson, C., Barth, A., Bortz, A., Shao, W., Boneh, D. Protecting browsers from DNS rebinding attacks. In Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS) (2007).
13. Jackson, C., Wang, H.J. Subspace: Secure cross-domain communication for web mashups. In Proceedings of the 16th International World Wide Web Conference (WWW) (2007).
14. De Keukelaere, F., Bhola, S., Steiner, M., Chari, S., Yoshihama, S. SMash: Secure cross-domain mashups on unmodified browsers. In Proceedings of the 17th International World Wide Web Conference (WWW) (2008). To appear.
15. Lowe, G. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In Proceedings of TACAS (volume 1055, 1996), Springer Verlag.
16. Microsoft. SECURITY attribute (FRAME, IFRAME). [http://msdn2.microsoft.com/en-us/library/ms534622\(VS.85\).aspx](http://msdn2.microsoft.com/en-us/library/ms534622(VS.85).aspx).
17. Needham, R.M., Schroeder, M.D. Using encryption for authentication in large networks of computers. *Commun. ACM*, 21, 12 (1978), 993–999.
18. Ross, D., January 2008. Personal communication.
19. Ruderman, J. *JavaScript Security: Same Origin*. <http://www.mozilla.org/projects/security/components/same-origin.html>.
20. Stuttard, D., Pinto, M. *The Web Application Hacker's Handbook*. Wiley, 2007.
21. Thorpe, D. Secure cross-domain communication in the browser. *Archit. J.* 12 (2007), 14–18.
22. Wang, H.J., Fan, X., Howell, J., Jackson, C. Protection and communication abstractions for web browsers in MashupOS. In Proceedings of the 21st ACM Symposium on Operating Systems Principles (SOSP) (2007).

### Adam Barth

(abarth@eecs.berkeley.edu),  
UC Berkeley.

### Collin Jackson

(collinj@cs.stanford.edu),  
Stanford University.

### John C. Mitchell

(mitchell@cs.stanford.edu),  
Stanford University.

© 2009 ACM 0001-0782/09/0600 \$10.00

# ACM Transactions on Internet Technology



This quarterly publication encompasses many disciplines in computing—including computer software engineering, middleware, database management, security, knowledge discovery and data mining, networking and distributed systems, communications, and performance and scalability—all under one roof. TOIT brings a sharper focus on the results and roles of the individual disciplines and the relationship among them. Extensive multi-disciplinary coverage is placed on the new application technologies, social issues, and public policies shaping Internet development.



<http://toit.acm.org/>

# Technical Perspective Software and Hardware Support for Deterministic Replay of Parallel Programs

By Norman P. Jouppi

**P**ARALLEL PROGRAMMING HAS long been recognized as a difficult problem. This problem has recently taken on a sense of urgency: the long march of single-thread performance increases has stopped in its tracks. Due to limitations on power dissipation and decreased return on investments in additional processor complexity, additional transistors provided by Moore's Law are now being channeled into a geometrically increasing number of cores per die. These cores can easily be applied to embarrassingly parallel problems and distributed computing in server environments by running multiple parallel tasks. However, there are many applications where increased performance on formerly single-threaded applications are highly desirable, ranging from personal computing devices to capability supercomputers.

A key problem in parallel programming is the ability to find concurrency bugs and to debug program execution in the presence of memory races on accesses to both synchronization and data variables. Subsequent executions of a parallel program containing a race or bug are unlikely to have the same exact ordering on each execution due to nondeterministic system effects. This may cause rare problems to occur long after deployment of an application. Rerunning programs in a slower debug mode also changes the relative timing, and can easily mask problems. Ideally what we'd like is a way to deterministically replay execution of parallel programs, by recording the outcome of memory races without significantly slowing down the execution of the original program. Additionally, one would like logging requirements of the execution to be manageable, and the replay of applications to occur at a speed similar to that of the original execution.

An important step in this direction appeared five years ago in the University of Wisconsin's Flight Data Recorder.<sup>1</sup> This original system required significant hardware state. However, for mainstream adoption, the additional hardware should be very small, since all users of a microprocessor design will be paying for the hardware support whether they use it or not. After several evolutionary enhancements in previous years, two systems appeared this year that make a quantum leap forward in reducing the overheads needed to support deterministic replay: instead of recording individual memory references, both of these systems only record execution of atomic blocks of instructions.

Rerun, also from the University of Wisconsin, reduces overhead by recording atomic episodes. An episode is a series of instructions from a single thread that happen to execute without conflicting with any other thread in the system. Episodes are created automatically by the recording system without

The following paper is a first for *Communications'* Research Highlights section: it contains a synthesis of recent work from two competing (but collegial) research teams.

modification of the applications. Rerun uses Lamport Scalar Clocks to order episodes and enable replay of an equivalent execution. Rerun reduces the hardware state per core to 166 bytes per core and the log size to only around 1.67 bytes/kiloinstruction per core in an 8-core system. This results in a  $\text{core}^*\log$  overhead product for many-core systems that is more than an order of magnitude smaller than previous work.

DeLorean, developed contemporaneously at the University of Illinois, executes large blocks of instructions atomically separated by checkpoints, like in transactional memory or thread-level speculation. Executing larger chunks of instructions provides benefits in both log size and replay speed. For example, for an 8-core processor, DeLorean is able to achieve a log size of only 0.0063 bytes/kiloinstruction per core while still being able to replay at 72% of the original execution speed. To put this in perspective, with this log size an entire day's execution of an 8-core processor would only take 20GB, a small fraction of a 1TB disk drive.

The following paper is a first for *Communications'* Research Highlights section: it contains a synthesis of recent work from two competing (but collegial) research teams. Both the Rerun and DeLorean teams were invited to contribute to this paper since their approaches appeared in the same conference session, both represent significant advances, and their approaches are actually complementary—Rerun requires very little additional hardware, whereas DeLorean can achieve much smaller log sizes but requires checkpoint and recovery hardware similar to that provided in transactional memory systems. Both research streams have the potential to make a significant impact on the productivity of future parallel programming. □

## Reference

- Xu, M., Bodik, R., and Hill, M.D. A "flight data recorder" for enabling full-system multiprocessor deterministic replay. *ACM/IEEE International Symposium on Computer Architecture*, June 2003.

**Norman P. Jouppi** (Norm.Jouppi@hp.com) is a Fellow and Director of Hewlett-Packard's Exascale Computing Lab in Palo Alto, CA.

# Two Hardware-Based Approaches for Deterministic Multiprocessor Replay

By Derek R. Hower, Pablo Montesinos, Luis Ceze, Mark D. Hill, and Josep Torrellas

## Abstract

Many shared-memory multithreaded executions behave nondeterministically when run on multiprocessor hardware such as emerging multicore systems. Recording nondeterministic events in such executions can enable deterministic replay—e.g., for debugging. Most challenging to record are memory races that can potentially occur on almost all memory references. For this reason, researchers have previously proposed hardware to record key memory race interactions among threads.

The two research groups coauthoring this paper independently uncovered a dual approach: *focus on recording how long threads execute without interacting*. From this common insight, the groups developed two significantly different hardware proposals. *Wisconsin Rerun* makes few changes to standard multicore hardware, while *Illinois DeLorean* promises much smaller log sizes and higher replay speeds. By presenting both proposals in one paper, we seek to illuminate the promise of the joint insight and inspire future designs.

## 1. INTRODUCTION

Modern computer systems are inherently nondeterministic due to a variety of events that occur during an execution, including I/O, interrupts, and DMA fills. The lack of repeatability that arises from this nondeterminism can make it difficult to develop and maintain correct software. Furthermore, it is likely that the impact of nondeterminism will only increase in the coming years, as commodity systems are now shared-memory multiprocessors. Such systems are not only impacted by the sources of nondeterminism in uniprocessors, but also by the outcome of memory races among concurrent threads.

In an effort to help ease the pain of developing software in a nondeterministic environment, researchers have proposed adding *deterministic replay* capabilities to computer systems. A system with a deterministic replay capability can record sufficient information during an execution to enable a replayer to (later) create an equivalent execution despite the inherent sources of nondeterminism that exist. With the ability to replay an execution verbatim, many new applications may be possible:

**Debugging:** Deterministic replay could be used to provide the illusion of a *time-travel debugger* that has the ability to selectively execute both forward and backward in time.

**Security:** Deterministic replay could also be used to enhance the security of software by providing the means for an in-depth analysis of an attack, hopefully leading to rapid patch deployment and a reduction in the economic impact of new threats.

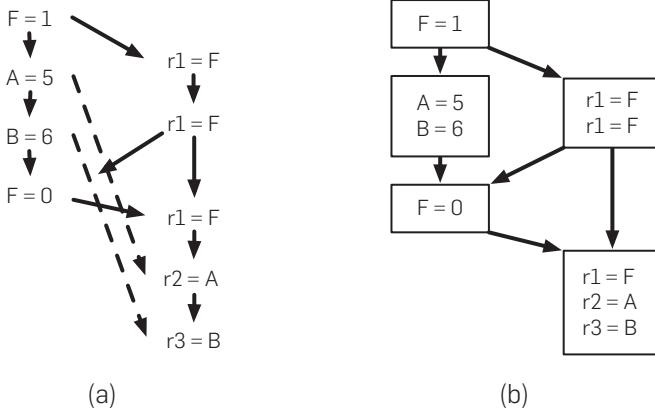
**Fault Tolerance:** With the ability to replay an execution, it may also be possible to develop hot-standby systems for critical service providers using commodity hardware. A virtual machine (VM) could, for example, be fed, in real time, the replay log of a primary server running on a physically separate machine. The standby VM could use the replay log to mimic the primary's execution, so that in the event that the primary fails, the backup can take over operation with almost zero downtime.

As existing commercial products have already shown, deterministic replay can be achieved with a software-only solution when executing in a uniprocessor environment.<sup>18</sup> This is due, in part, to the fact that sources of nondeterminism in a uniprocessor, such as interrupts or I/O, are relatively rare events that take a long time to complete. However, when executing in a shared-memory multiprocessor environment, memory races, which can potentially occur on every memory access, are another source of nondeterminism. All-software solutions exist,<sup>4, 8</sup> but results show that they do not perform well on workloads that interact frequently. Thus, it is likely that a general solution will require hardware support. To this end, Bacon and Goldstein<sup>2</sup> originally proposed recording all snooping coherence transactions, which, while fast, produced a serial and voluminous log (see Figure 1).

Xu et al.<sup>16</sup> modernized hardware support for multiprocessor deterministic replay in general and *memory race recording* in particular. A memory race recorder is responsible for logging enough information to reconstruct the order of all fine-grained memory interleavings that occur during an execution. To reduce the amount of information that needs to be logged (so that longer periods can be recorded for a fixed hardware cost), the system proposed by Xu et al. implemented in hardware an enhancement to Netzer's transitive reduction optimization.<sup>13</sup> The idea is to skip the logging of those races that can be implied through transitivity, i.e., those races

The original Wisconsin Rerun<sup>6</sup> paper as well as the original Illinois DeLorean<sup>11</sup> paper were published in the *Proceedings of the 35th Annual International Symposium on Computer Architecture* (June 2008).

**Figure 1: An example of efficient race recording using (a) an explicit transitive reduction and (b) independent regions.** In (a), solid lines between threads are races written to the log, while dashed lines are those races implied through transitivity.



implied through the combination of previously logged races and sequential program semantics. Figure 1a illustrates a transitive reduction. Inter-thread races between instructions accessing locations A and B, respectively, are not logged since they are implied by the recorded race for location F.

While both the original<sup>16</sup> and follow-on<sup>17</sup> work by Xu et al. were successful in achieving efficient log compression (~1B/1000 instructions executed), they required a large amount of hardware state, on the order of an additional L1 cache per core, in order to do so. Subsequent work by Narayanasamy et al.<sup>12</sup> on the Strata race recorder reduced this hardware requirement but, as results in Hower and Hill<sup>6</sup> show, may not scale well as the number of hardware contexts in a system increases. This is largely because Strata writes global information to its log entries that contains a component from each hardware thread context in the system.

A key observation, discovered independently by the authors of this paper at the Universities of Illinois and Wisconsin, is that by focusing on regions of *independence*, rather than on individual dependencies, an efficient and scalable memory race recorder can be made *without* sacrificing logging efficiency. Figure 1b illustrates this notion by breaking the execution of Figure 1a into an ordered series of independent execution regions. Because intra-thread dependencies are implicit and do not need to be recorded, the execution in Figure 1b can be completely described by the three inter-thread dependencies, which is the same amount of information required after a transitivity reduction shown in Figure 1a.

The authors of this paper have developed two different systems, called *Rerun*<sup>6</sup> and *DeLorean*,<sup>11</sup> that both exploit the same independence observation described above. These systems, presented in the same session of ISCA 2008, exemplify different trade-offs in terms of logging efficiency and implementation complexity. Rerun can be implemented with small modifications to existing memory system architectures but writes a larger log than DeLorean. DeLorean can achieve a greater log size reduction and a higher replay speed but requires novel hardware to do so.

## 2. RERUN

Wisconsin Rerun<sup>6</sup> exploits the concept of *episodic race recording* to achieve efficient logging with only small modifications to existing memory system architectures. The Rerun race recorder does not interfere with a running program in any way; it is an impartial observer of a running execution, and as such avoids artificially perturbing the execution under observation.

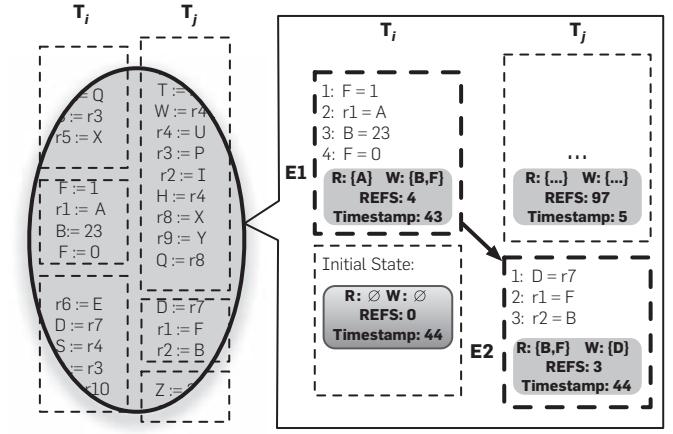
### 2.1. Episodic memory race recording

This section develops insights behind Rerun. It motivates Rerun with an example, gives key definitions, and explains how Rerun establishes and orders episodes.

**Motivating Example and Key Ideas:** Consider the execution in Figure 2 that highlights two threads *i* and *j* executing on a multicore system. Dynamic instructions 1–4 of thread *i* happen to execute without interacting with instructions running concurrently on thread *j*. We call these instructions, collectively labeled  $E_1$ , an episode in thread *i*'s execution. Similarly, instructions 1–3 of thread *j* execute without interaction and constitute an episode  $E_2$  for thread *j*. As soon as a thread's episode ends, a new episode begins. Thus, every instruction execution is contained in an episode, and episodes cover the entire execution (right side of Figure 2).

Rerun must solve two subproblems in order to ensure that enough episodic information is recorded to enable deterministic replay of all memory races. First, it must determine when an episode ends, and, by extension, when the next one begins. To remain independent, an episode  $E$  must end when another thread issues a memory reference that *conflicts* with references made in episode  $E$ . Two memory accesses conflict if they reference the same memory block, are from different threads, and at least one is a write. For example, episode  $E_1$  in Figure 2 ends because thread *j* accesses the variable *F* that was previously written (i.e., *F* is in the write set of  $E_1$ ). Formally, for all combinations of episodes  $E$  and  $F$

**Figure 2: An example of episodic recording.** Dashed lines indicate episode boundaries. In the blown up diagram of threads *i* and *j*, the shaded boxes show the state of the episode as it ends, including the read and write sets, memory reference counter, and the timestamp. The shaded box in the last episode of thread *i* shows the initial episode state.



in an execution, the *no-conflict* condition of Equation 1 must hold. Let  $R_E(W_E)$  denote episode  $E$ 's read (write) set:

$$[W_E \cap (R_F \cup W_F) = \emptyset] \wedge [R_E \cap W_F = \emptyset] \quad (1)$$

Importantly, while an episode *must* end to avoid conflicts, episodes *may* end early for any or no reason. In Section 2.2, we will ease implementation cost by ending some episodes early.

Second, an episodic recorder must establish an ordering of episodes among threads. Rerun does so using Lamport scalar clocks,<sup>7</sup> which is a technique that guarantees the timestamp of any episode  $E$  executing on thread  $i$  has a scalar value that is greater than the timestamp of any episode on which  $E$  is dependent and less than the timestamp of any episode dependent on  $E$ . In our example, since the episode  $E_1$  ends with a timestamp of 43, the subsequent episode executing on thread  $j$  ( $E_2$ ), which uses block  $F$  after thread  $i$ , must be assigned a timestamp of (at least) 44.

The specific Rerun mechanism meets three conditions sufficient for a Lamport scalar clock implementation:

1. When an episode  $E$  on  $thread_E$  begins, its  $timestamp_E$  begins with a value one greater than the timestamp of the previous episode executed by  $thread_E$  (or 0 if episode  $E$  is  $thread_E$ 's first episode).
2. When an episode  $E$  adds a block to its read set  $R_E$  that was most-recently in the write set  $W_D$  of completed episode  $D$ , it sets its  $timestamp_E$  to  $\max(timestamp_E, timestamp_{D+1})$ .
3. When an episode  $E$  adds a block to its write set  $W_E$  that was most-recently in the write set  $W_{D_0}$  of completed episode  $D_0$  or in the read set of any episode  $D_1 \dots D_N$ , it sets its  $timestamp_E$  to  $\max(timestamp_E, timestamp_{D_0} + 1, timestamp_{D_1} + 1, \dots, timestamp_{D_N} + 1)$ .

When each episode  $E$  ends, Rerun logs both  $timestamp_E$  and  $references_E$  in a per-thread log.  $references_E$  is a count of memory references completed in  $E$ , and is used to record the episode length. The Lamport clock algorithm ensures that the execution order of all conflicting episodes corresponds to monotonically increasing timestamps. Two episodes can only be assigned the same timestamp if they do not conflict

and, thus, can be replayed in any alternative order without affecting replay fidelity.

A replayer (not shown) uses information about episode duration and ordering to reconstruct an execution with the same behavior. If episodes are replayed in timestamp order, then the replayed execution will be logically equivalent to the recorded execution. Unfortunately, the use of Lamport scalar clocks make Rerun's replay (mostly) sequential.

## 2.2. Rerun implementation

Here we develop a Rerun implementation for a system based on a cache-coherent multicore chip, with key parameters shown in Table 1. Though we describe Rerun in terms of a specific base system, the mechanism can be extended to other systems, including those with a TSO memory consistency model, out-of-order cores, multithreaded cores, alternate cache designs, and snooping coherence. Details of the changes needed to accommodate these alternate architectures can be found in the original paper.<sup>6</sup>

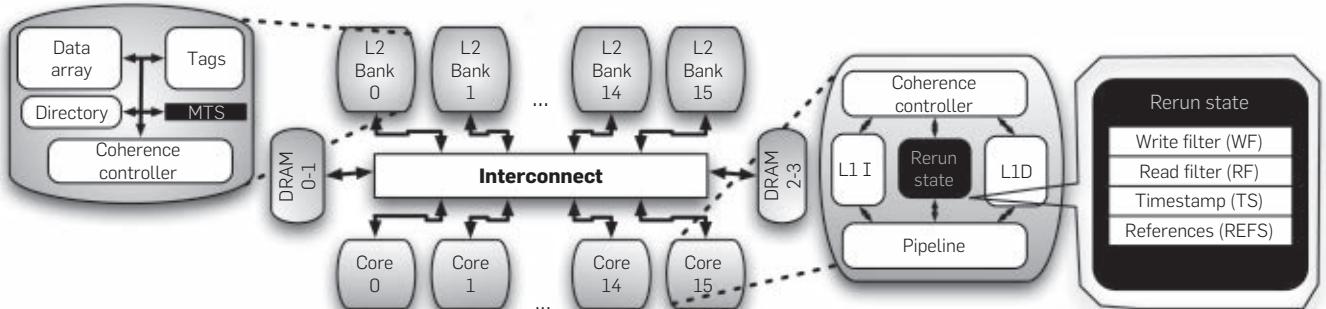
**Rerun Hardware:** As Figure 3 depicts, Rerun adds modest hardware state to the base system. To each core, Rerun adds:

- Read and Write Bloom filters,  $WF$  and  $RF$ , to track the current episode's write and read sets (e.g., 32B and 128B, respectively).
- A Timestamp Register,  $TS$ , to hold the Lamport Clock of the current episode executing on the core (e.g., 4B).
- A Memory Reference Counter,  $REFS$ , to record the current episode's references (e.g., 2B).

**Table 1: Base system configuration.**

Cores	16, in-order, 3GHz
L1 Caches	Split I&D, private, 32K four-way set associative, write-back, 64B lines, LRU replacement, three cycle hit
L2 Caches	Unified, shared, inclusive, 8M 8-way set associative, write-back, 16 banks, LRU replacement, 37 cycle hit
Directory	Full bit vector in L2
Memory	4G DRAM, 300 cycle access
Coherence	MESI directory, silent replacements
Consistency Model	Sequential consistency (SC)

**Figure 3: Rerun hardware.**



To each L2 cache bank, Rerun also adds a “memory” timestamp register, *MTS* (e.g., 4B). This register holds the maximum of all timestamps for victimized blocks that map to its bank. A victimized block is one replaced from an L1 cache, and its timestamp is the timestamp of the core at the time of victimization.

Finally, coherence response messages—data, acknowledgements, and writebacks—carry logical timestamps. Book-keeping state, such as a per-core pointer to the end of its log, is not shown.

**Rerun Operation:** During execution, Rerun monitors the no-conflict equation by comparing the addresses of incoming coherence requests to those in *RF* and *WF*. When a conflict is detected, Rerun writes the tuple  $\langle TS, REFS \rangle$  to a per-thread log, then begins a new episode by resetting *REFS*, *WF*, and *RF*, and by incrementing the local timestamp *TS* according to the algorithm in Section 2.1.

By gracefully handling virtualization events, Rerun allows programmers to view logs as *per thread*, rather than *per core*. At a context switch, the OS ends the core’s current episode by writing *REFS* and *TS* state to the log. When the thread is rescheduled, it begins a new episode with reset *WF*, *RF*, and *REFS*, and a timestamp equal to the max of the last logged *TS* for that thread and the *TS* of the core on which the thread is rescheduled. Similarly, Rerun can handle paging by ensuring that TLB shootdowns end episodes.

Rerun also ends episodes when implementation resources are about to be exhausted. Ending episodes just before 64K memory references, for example, allows *REFS* to be logged in 2B.

### 2.3. Evaluation

**Methods:** We evaluate the Rerun recording system using the Wisconsin GEMS<sup>10</sup> full system simulation infrastructure. The simulator configuration matches the baseline shown in Table 1 with the addition of Rerun hardware support. Experiments were run using the Wisconsin Commercial Workload Suite.<sup>1</sup> We tested Rerun with these workloads and a microbenchmark, racey, that uses number theory to produce an execution whose outcome is highly sensitive to memory race ordering (available at [www.cs.wisc.edu/~markhill/racey.html](http://www.cs.wisc.edu/~markhill/racey.html)).

**Rerun Performance:** Figure 4 shows the performance of Rerun on all four commercial workloads. Rerun achieves an uncompressed log size of about 4B logged per 1000 instructions. Importantly, we notice modest variation among the log size of each workload, leading us to believe that Rerun can perform well under a variety of memory access patterns.

We show the relative performance of Rerun in comparison to the prior state of the art in memory race recording in Figure 5. Rerun achieves a log size comparable to the most efficient prior recorder (RTR<sup>17</sup>), but does so with a fraction of the hardware cost (~0.2KB per core vs. 24KB per core). Like RTR, and unlike Strata,<sup>12</sup> Rerun scales well as the number of cores in the system increases, due, in part, to the fact that Rerun and RTR both write thread-local log entries rather than a global entry with a component from each thread.

Figure 4: Rerun absolute log size.

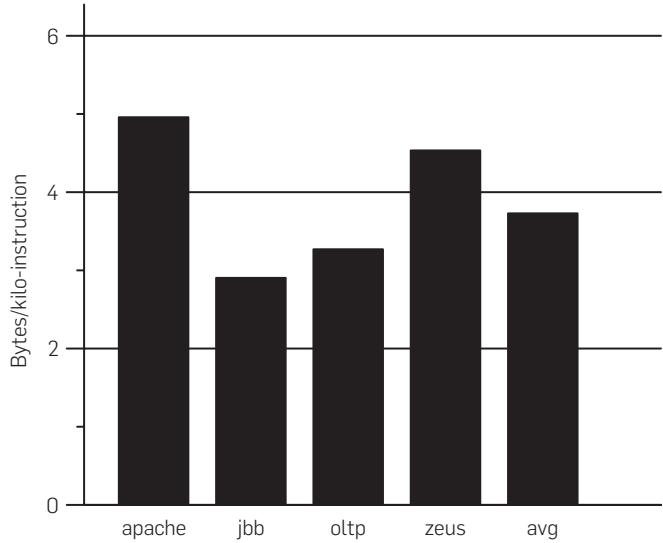
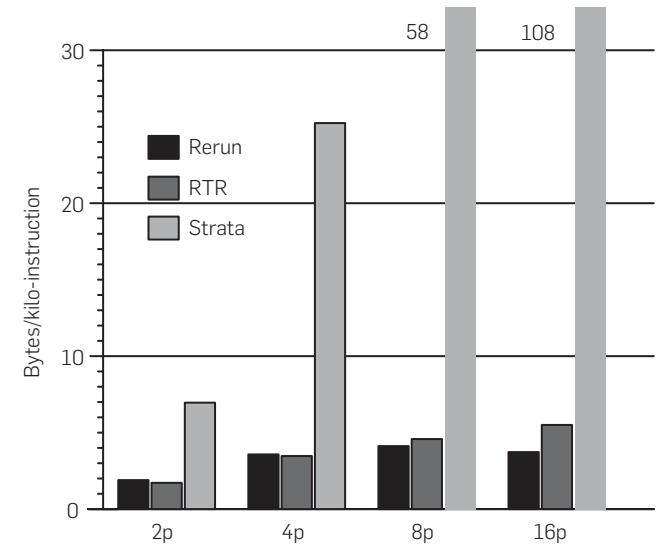


Figure 5: Hardware cost comparison to RTR and Strata.



### 3. DELOREAN

Illinois DeLorean<sup>11</sup> is a new approach to deterministic replay that exploits the opportunities afforded by a new execution substrate: one where processors continuously execute large blocks of instructions atomically, separated by register checkpoints.<sup>3, 5, 9, 15</sup> In this environment, to capture a multi-threaded execution for deterministic replay, DeLorean only needs to log the total *order* in which blocks from different processors commit.

This approach has several advantages. First, it results in a substantial reduction in log size compared to previous schemes—at least about one order of magnitude. Second, DeLorean can replay at a speed comparable to that of the initial execution. Finally, in an aggressive operation mode, where DeLorean predefines the commit order of the blocks

from different processors, DeLorean generates only a very tiny log—although there is a performance cost. While *DeLorean*'s execution substrate is not standard in today's hardware systems, the required changes are mostly concentrated in the memory system.

### 3.1. The DeLorean idea

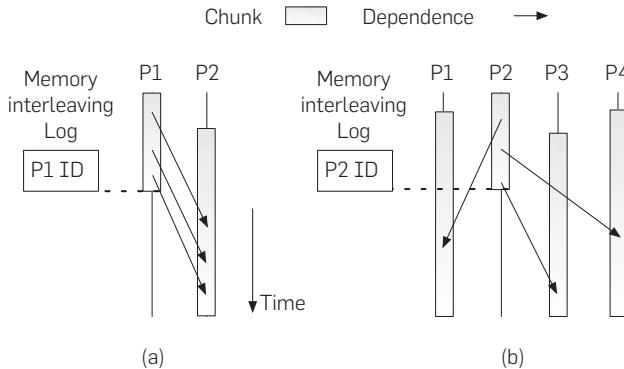
There have been several proposals for multiprocessors where processors continuously execute blocks of consecutive dynamic instructions atomically and in isolation.<sup>3, 5, 9, 15</sup> In this environment, the updates made by a block of instructions (or *Chunk*) only become visible when the chunk commits. When two chunks running concurrently on two different processors conflict—there is a data dependence across the two chunks—the hardware typically squashes and retries one of the chunks. Moreover, after a chunk completes execution, there is an optimized global commit step in an arbiter module that informs the relevant processors that the chunk is committed. The net effect is that the interleaving between the memory accesses of different processors appears to occur *only* at chunk boundaries.

In such environment, recording the execution for replay simply involves logging the total sequence of chunk commits. This has two very important consequences for replay systems. The first one is that the memory ordering log is now very small. Indeed, rather than recording individual dependences or groups of them like in all past proposals, the log in a chunk-based system only needs to record the *total order* in which chunks from different processors commit. This means that each log entry is short (the ID of the committing processor, if all chunks have the same size), and that the log is updated infrequently (chunks are thousands of instructions long).

The second consequence is that, because the memory accesses issued by a processor inside a chunk are not visible to the rest of the processors until the chunk commits, such accesses can be fully reordered and overlapped. This means that both execution and replay under DeLorean proceed at a high speed.

DeLorean naturally combines multiple data dependences between two or more processors into a single entry in the log that records the memory interleaving—the Memory Interleaving Log. An example is shown in Figure 6a, where

**Figure 6: Combining multiple dependences into a single log entry.**



all the dependences between the accesses in the chunks executed by processors *P1* and *P2* (shown with arrows in the figure) are combined into a *single* entry in the log. The figure also shows that such log entry is *simply* *P1*'s ID. In a second example shown in Figure 6b, multiple dependences across several processors are summarized in a single log entry. Specifically, the *single* log entry inserted when the chunk from *P2* commits is enough to summarize the three dependences.

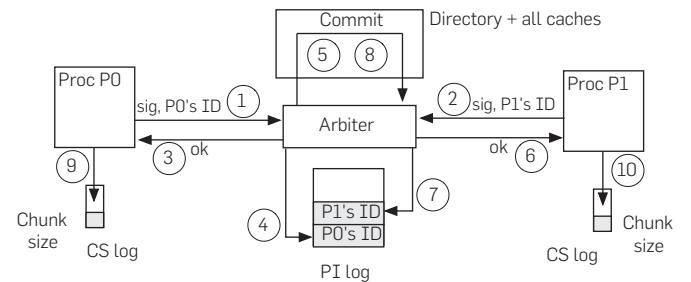
### 3.2. DeLorean execution modes

DeLorean provides two main execution modes, namely *OrderOnly* and *PicoLog*. To understand them, we start by describing a naive, third execution mode called *Order&Size*. In *Order&Size*, each log entry contains the ID of the processor committing the chunk and the chunk size—measured in number of retired instructions. During execution, an arbiter module (a simple state machine that enforces chunk commit order<sup>3</sup>) logs the sequence of committing processor IDs in a *Processor Interleaving* (PI) log. At the same time, processors record the size of the chunk they commit in a per-processor *Chunk Size* (CS) log. The combination of a single PI log and per-processor CS logs constitutes the Memory Interleaving Log.

Figure 7 shows DeLorean's operation in *Order&Size* mode. During the initial execution, when a processor such as *P0* or *P1* finishes a chunk, it sends a request-to-commit message to the arbiter (steps 1 and 2). Such messages contain the processor IDs plus Bloom-filter signatures that summarize the memory footprint of the chunks<sup>3</sup> (*sig* in the figure). Suppose that the arbiter grants permission to *P0* first (step 3). In this case, the arbiter logs *P0*'s ID (4) and propagates the commit operation to the rest of the machine (5). While this is in progress, if the arbiter determines that both chunks can commit in parallel, it sends a commit grant message to *P1* (6), logs *P1*'s ID (7), and propagates the commit (8). As each processor receives commit permission, it logs the chunk size (9 and 10).

Our first DeLorean execution mode, called *OrderOnly*, omits logging chunk sizes by making “chunking”—i.e., the decision of when to finish a chunk—deterministic. DeLorean accomplishes this by finishing chunks when a fixed number of instructions have been committed. In reality, certain events truncate a currently running chunk and force it to commit before it has reached its “expected” size. This is fine as long as the event reappears deterministically in the replay. For example, consider an uncached load to an I/O port. The chunk is truncated but its log entry does not

**Figure 7: DeLorean's operation.**



need to record its actual size because the uncached load will reappear in the replay and truncate the chunk at the same place. There are, however, a few events that truncate a currently running chunk and are not deterministic. When one such event occurs, the CS log adds an entry with: (1) what chunk gets truncated (its position in the sequence of chunks committed by the processor) and (2) its size. With this information, the exact chunking can be reproduced during replay.

Consequently, *OrderOnly* generates a PI log with only processor IDs and very small per-processor CS logs. For the large majority of chunks, steps 9 and 10 in Figure 7 are skipped.

Our second DeLorean execution mode, called *PicoLog*, builds on *OrderOnly* and additionally eliminates the need for a PI log by “predefining” the chunk commit interleaving during both initial execution and replay. This is accomplished by enforcing a given commit policy—e.g., pick processors round-robin, allowing them to commit one chunk at a time. It needs only the tiny per-processor CS log discussed for *OrderOnly*. Thus, the *Memory Interleaving Log* is largely eliminated. The drawback is that, by delaying the commit of completed chunks until their turn, *PicoLog* may slow down execution and replay.

Looking at Figure 7, *PicoLog* skips steps 4, 7 and, typically, 9 and 10. The arbiter grants commit permission to processors according to a predefined order policy, irrespective of the order in which it receives their commit requests. Note, however, that a processor does not stall when requesting commit permission; it continues executing its next chunk(s).<sup>3</sup>

Table 2 shows the PI and CS logs in each of the two execution modes and *Order&Size*.

### 3.3. DeLorean implementation

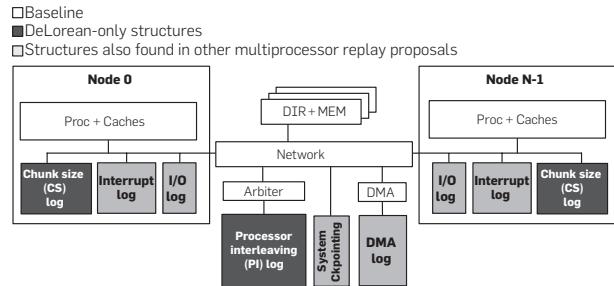
Our DeLorean implementation uses a machine that supports a chunk-based execution environment with a generic network and an arbiter. It augments it with the three typical mechanisms for replay: the Memory Interleaving Log (consisting of the PI and CS logs), the input logs, and system checkpointing (Figure 8).

The input logs are similar to those in previous replay schemes. As shown in Figure 8, they include one shared log (*DMA log*) and two per-processor logs (*Interrupt* and *I/O* logs). The DMA acts like another processor in that, before it updates memory, it needs to get commit permission from the arbiter. Once permission is granted, the DMA log logs the data that the DMA writes to memory. The per-processor

**Table 2: PI and CS logs in each execution mode.**

Execution Mode	PI Log		CS Log	
	Log Entry Format	When Updated	Log Entry Format	When Updated
<i>Order&amp;Size</i>	procID	Chunk commit	size	Chunk commit
<i>OrderOnly</i>	procID	Chunk commit	chunkID, size	Chunk truncation
<i>PicoLog</i>	—	—	chunkID, size	Chunk truncation

**Figure 8: Overall DeLorean system implementation.**



Interrupt log stores, for each interrupt, the time it is received, its type, and its data. Time is recorded as the processor-local chunk ID of the chunk that initiates execution of the interrupt handler. The per-processor I/O log records the values obtained by I/O loads. Like in previous replay schemes, DeLorean includes system checkpointing support.

### 3.4. DeLorean replay

During replay, processors must execute the same chunks and commit them in the same order. In *Order&Size*, each processor generates chunks that are sized according to its CS log, while in *OrderOnly* and *PicoLog*, processors use the CS log only to recreate the chunks that were truncated nondeterministically. In *Order&Size* and *OrderOnly*, the arbiter enforces the commit order present in the PI log.

As an example, consider the log generated during initial execution as shown in Figure 7. During replay, suppose that *P1* finishes its chunk before *P0*, and the arbiter receives message 2 before 1. The arbiter checks its PI log (or its predefined order policy in *PicoLog*) and does not grant permission to commit to *P1*. Instead, it waits until it receives the request from *P0* (message 1). At that point, it grants permission to commit to *P0* (3) and propagates its commit (5). The rest of the operation is as in the initial execution but without logging. In addition, processors use their CS log to decide when to finish each chunk (*Order&Size*) or those chunks truncated nondeterministically during the initial execution (*OrderOnly* and *PicoLog*).

Thanks to our chunk-based substrate, during replay all processors execute concurrently. Moreover, each processor fully reorders and overlaps its memory accesses within a chunk. Chunk commit involves a fast check with the arbiter.<sup>3</sup> The processor overlaps such check with the computation of its next chunk.

### 3.5. Exceptional events

In DeLorean, the same instruction in the initial and the replayed execution must see exactly the same full-system architectural state. On the other hand, it is likely that structures that are not visible to the software such as the cache and branch predictor will contain different state in the two runs.

Unfortunately, chunk construction is affected by the cache state—through cache overflow that requires finishing the chunk—and by the branch predictor—through wrong-path speculative loads that may cause spurious dependences

**Table 3: Exceptional events that may affect chunk construction.**

Do Not Truncate a Chunk	Truncate a Chunk	
	Deterministically	Nondeterministically
1. Interrupts	1. Reach limit number of instructions	1. Cache overflow attempt
2. Traps	2. Uncached accesses (e.g., I/O initiation) 3. Special system instructions	2. Repeated chunk collision

and induce chunk squashes. Consequently, we need to be careful that chunks are still replayed deterministically.

Table 3 lists the exceptional events that might affect chunk construction during the initial execution. A full description of these events and the actions taken when they occur is presented in Montesinos et al.<sup>11</sup> At a high level, there are events that do not truncate the chunk, events that truncate it deterministically, and events that truncate it nondeterministically. The latter are the only ones that induce the logging of an entry in the CS log. Such events are the attempt to overflow the cache and repeated chunk collision. Overall, as described in Montesinos et al.,<sup>11</sup> even in the presence of all these types of exceptional events, DeLorean’s replay is deterministic.

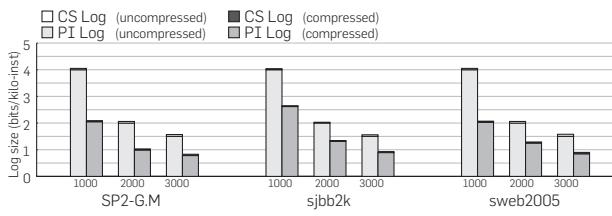
### 3.6. Evaluation

We used the SESC simulator<sup>14</sup> to evaluate DeLorean. We simulated a chip multiprocessor with eight cores clocked at 5GHz. We ran the SPLASH-2 applications as well as SPECjbb2000 and SPECweb2005. In our evaluation, we estimated DeLorean’s log size and its performance during recording and replay. In this section, we show a summary of the evaluation presented in Montesinos et al.<sup>11</sup>

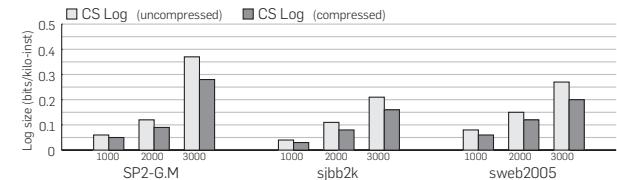
Figure 9 shows the size of the PI and CS logs in *OrderOnly* in bits per kilo-instruction. We evaluate DeLorean configurations with standard chunk sizes of 1,000, 2,000, and 3,000 instructions. For each of them, we report the size of both logs with and without compression. In the figure, the CS log contribution is stacked atop the PI log’s. The SP2-G.M. bars correspond to the geometric mean of SPLASH-2.

The figure shows that our preferred 2,000-inst. *OrderOnly* configuration uses on average only 2.1b (or 1.3b if compressed) per kilo-instruction to store both the PI and CS logs. For comparison purposes, the estimated average size of the compressed Memory Races Log in RTR under Sequential

**Figure 9: Size of the PI and CS logs in *OrderOnly*. The numbers under the bars are the standard chunk sizes in instructions.**



**Figure 10: Size of the CS log in *PicoLog*.** Recall that *PicoLog* has no PI log. The numbers under the bars are the standard chunk sizes in instructions.



Consistency (SC) from Xu et al.<sup>17</sup> is 8b per kilo-instruction. We call this system Basic RTR and use it as a reference, although we note that the set of applications measured here and in Xu et al.<sup>17</sup> are different. This means that these compressed logs use only 16% of the space that we estimate is needed by the compressed Memory Races Log in Basic RTR.

Figure 10 shows the size of the CS log in *PicoLog*. Recall that *PicoLog* has no PI log. We see that the CS log needs 0.37b or fewer per kilo-instruction in all cases—even without compression. Our preferred 1,000-instruction *PicoLog* configuration generates a compressed log with an average of only 0.05b per kilo-instruction. To put this in perspective, it implies that, if we assume an IPC of 1, the combined effect of all eight 5GHz processors is to produce a log of only about 20GB per day.

Finally, we consider the speed of DeLorean during recording and replay. It can be shown that *OrderOnly* introduces negligible overhead during recording, and that it enables replay, on average, at 82% of the recording speed. Under *PicoLog*, recording and replay speeds decrease, on average, to 86% and 72%, respectively, of the recording speed under *OrderOnly*.

## 4. CONCLUSION

This paper presented two novel hardware-based approaches for deterministic replay of multiprocessor executions, namely *Wisconsin Rerun* and *Illinois DeLorean*. Both approaches seek to enable deterministic replay by focusing on recording how long threads execute without interacting. Rerun makes few changes to standard multicore hardware, while DeLorean promises much smaller log sizes and higher replay speeds. Future work includes improving Rerun’s replay speed, generalizing DeLorean’s hardware design alternatives, and making the original multithreaded executions more deterministic.

## Acknowledgments

We thank Norman Jouppi and David Patterson for suggesting this article and Norman Jouppi for writing the Perspective. Hower and Hill thank those acknowledged in the Rerun paper, including NSF grants CCR-0324878, CNS-0551401, and CNS-0720565. Hill has a significant financial interest in Sun Microsystems. Montesinos, Ceze, and Torrellas acknowledge the support provided by NSF under grants CCR-0325603 and CNS-0720593 and Intel and Microsoft for funding this work under the Universal Parallel Computing Research Center. □

## References

1. Alameldeen, A.R., Mauer, C.J., Xu, M., Harper, P.J., Martin, M.M.K., Sorin, D.J., Hill, M.D., Wood, D.A. Evaluating non-deterministic multi-threaded commercial workloads. In *Proceedings of the 5th Workshop on Computer Architecture Evaluation Using Commercial Workloads* (February 2002), 30–38.
2. Bacon, D.F., Goldstein, S.C. Hardware-assisted replay of multiprocessor programs. *Proceedings of the ACM/ONR Workshop on Parallel and Distributed Debugging*, published in ACM SIGPLAN Notices (1991), 194–206.
3. Ceze, L., Tuck, J.M., Montesinos, P., Torrellas, J. BulkSC: Bulk Enforcement of Sequential Consistency. In *Proceedings of the 34th International Symposium on Computer Architecture* (San Diego, CA, USA, June 2007).
4. Dunlap, G.W., Lucchetti, D., Chen, P.M., Fetterman, M. Execution replay on multiprocessor virtual machines. In *International Conference on Virtual Execution Environments (VEE)* (2008).
5. Hammond, L., Wong, V., Chen, M., Carlstrom, B.D., Davis, J.D., Hertzberg, B., Prabhu, M.K., Wijaya, H., Kozyrakis, C., Olukotun, K. Transactional memory coherence and consistency. In *Proceedings of the 34th International Symposium on Computer Architecture* (June 2004).
6. Hower, D.R., Hill, M.D. Rerun: Exploiting episodes for lightweight race recording. In *Proceedings of the 35th Annual International Symposium on Computer Architecture* (June 2008).
7. Lampert, L. Time, clocks and the ordering of events in a distributed system. *Commun. ACM* 21, 7 (July 1978), 558–565.
8. Leblanc, T.J., Mellor-Crummey, J.M. Debugging parallel programs with instant replay. *IEEE Trans. Comp. C-36*, 4 (April 1987), 471–482.
9. Lucia, B., Devietti, J., Strauss, K., Ceze, L. Atom-aid: Detecting and surviving atomicity violations. In *Proceedings of the 35th International Symposium on Computer Architecture* (June 2008).
10. Martin, M.M.K., Sorin, D.J., Beckmann, B.M., Marty, M.R., Xu, M., Alameldeen, A.R., Moore, K.E., Hill, M.D., Wood, D.A. Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset. *Comp. Arch. News* (September 2005), 92–99.
11. Montesinos, P., Ceze, L., Torrellas, J. DeLorean: Recording and deterministically replaying shared-memory multiprocessor execution efficiently. In *Proceedings of the 35th International Symposium on Computer Architecture* (June 2008).
12. Narayanasamy, S., Pereira, C., Calder, B. Recording shared memory dependencies using strata. In *Proceedings of the 12th International Conference*
- on Architectural Support for Programming Languages and Operating Systems (New York, NY, USA, October 2006), 229–240.
13. Netzer, R.H.B. Optimal tracing and replay for debugging shared-memory parallel programs. In *Workshop on Parallel and Distributed Debugging* (San Diego, California, May 1993), 1–11.
14. Renau, J., Fraguerau, B., Tuck, J., Liu, W., Prvulovic, M., Ceze, L., Sarangi, S., Sack, P., Strauss, K., Montesinos, P. SESC Simulator (January 2005), <http://sesc.sourceforge.net>.
15. Vallejo, E., Galluzzi, M., Cristal, A., Vallejo, F., Beivide, R., Stenstrom, P., Smith, J.E., Valero, M. Implementing kilo-instruction multiprocessors. In *Proceedings of the 2005 International Conference on Pervasive Systems* (July 2005).
16. Xu, M., Bodik, R., Hill, M.D. A "flight

**Derek R. Hower** (drh5@cs.wisc.edu)  
Computer Sciences Department  
University of Wisconsin-Madison.

**Pablo Montesinos** (pmontesi@cs.uiuc.edu)  
Computer Science Department  
University of Illinois  
Urbana-Champaign.

**Luis Ceze** (luisceze@cs.washington.edu)  
Department of Computer Science  
and Engineering  
University of Washington.

"data recorder" for enabling full-system multiprocessor deterministic replay. In *Proceedings of the 30th Annual International Symposium on Computer Architecture* (June 2003), 122–133.

17. Xu, M., Bodik, R., Hill, M.D. A regulated transitive reduction (RTR) for longer memory race recording. In *Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems* (October 2006), 49–60.

18. Xu, M., Malyugin, V., Sheldon, J., Venkitachalam, G., Weissman, B. Retrace: Collecting execution trace with virtual machine deterministic replay. In *Proceedings of the 3rd Annual Workshop on Modeling, Benchmarking and Simulation* (June 2007).

**Mark D. Hill** (markhill@cs.wisc.edu)  
Computer Sciences Department  
University of Wisconsin-Madison.

**Josep Torrellas** (torrellas@cs.uiuc.edu)  
Computer Science Department  
University of Illinois  
at Urbana-Champaign.

© 2009 ACM 0001-0782/09/0600 \$10.00

## Take Advantage of ACM's Lifetime Membership Plan!

- ◆ **ACM Professional Members** can enjoy the convenience of making a single payment for their entire tenure as an ACM Member, and also be protected from future price increases by taking advantage of **ACM's Lifetime Membership** option.
- ◆ **ACM Lifetime Membership** dues may be tax deductible under certain circumstances, so becoming a Lifetime Member can have additional advantages if you act before the end of 2009. (Please consult with your tax advisor.)
- ◆ Lifetime Members receive a certificate of recognition suitable for framing, and enjoy all of the benefits of **ACM Professional Membership**.

Learn more and apply at:  
<http://www.acm.org/life>



Association for  
Computing Machinery

Advancing Computing as a Science & Profession

## Expansion of the Research School „Service-Oriented Systems Engineering“ at Hasso-Plattner-Institute

8 Ph.D. grants available - starting October 1, 2009

Hasso-Plattner-Institute (HPI) is a privately financed institute affiliated with the University of Potsdam, Germany. The Institute's founder and benefactor Professor Hasso Plattner, who is also co-founder and chairman of the supervisory board of SAP AG, has created an opportunity for students to experience a unique education in IT systems engineering in a professional research environment with a strong practice orientation.

In 2005, HPI initiated the research school in „Service-Oriented Systems Engineering“ under the scientific supervision of Professors Jürgen Döllner, Holger Giese, Robert Hirschfeld, Christoph Meinel, Felix Naumann, Hasso Plattner, Andreas Polze, Mathias Weske and Patrick Baudisch.

We are expanding our research school and are currently seeking

### **8 Ph.D. students (monthly stipends 1400 - 1600 Euro) 2 Postdocs (monthly stipend 1800 Euro)**

Positions will be available starting October 1, 2009. The stipends are not subject to income tax.

**The main research areas in the research school at HPI are:**

- Self-Adaptive Service-Oriented Systems
- Operating System Support for Service-Oriented Systems
- Architecture and Modeling of Service-Oriented Systems
- Adaptive Process Management
- Services Composition and Workflow Planning
- Security Engineering of Service-Based IT Systems
- Quantitative Analysis und Optimization of Service-Oriented Systems
- Service-Oriented Systems in 3D Computer Graphics
- Service-Oriented Geoinformatics

**Prospective candidates are invited to apply with:**

- Curriculum vitae and copies of degree certificates/transcripts
- A short research proposal
- Writing samples/copies of relevant scientific papers (e.g. thesis, etc.)
- Letters of recommendation

Please submit your applications before August 15, 2009 to the coordinator of the research school:

**Prof. Dr. Andreas Polze**  
**Hasso-Plattner-Institute, Universität Potsdam**  
Postfach 90 04 60, 14440 Potsdam, Germany

Successful candidates will be notified by September 15, 2009 and are expected to enroll into the program on October 1, 2009.

For additional information see:

<http://kolleg.hpi.uni-potsdam.de> or contact the office:  
Telephone +49-331-5509-220, Telefax +49-331-5509-229  
Email: [office-polze@hpi.uni-potsdam.de](mailto:office-polze@hpi.uni-potsdam.de)



## Frostburg State University

Assistant Professor of Computer Science

Frostburg State University, Computer Science Department seeks applications for a full-time tenure track Assistant Professor of Computer Science to begin in Fall 2009. Salary commensurate with experience and includes USM benefits package. For more information, visit [www.frostburg.edu/hr/jobs.htm](http://www.frostburg.edu/hr/jobs.htm). EEO

---

## Kansas State University

Research Fellow - Computing  
and Information Sciences

The KDD Lab at Kansas State University has an opening for a research fellow. The ideal candidate possesses research experience in the areas of information retrieval, information extraction, natural language processing, and/or visualization. Screening begins May 4, 2009 and continues until the position is filled. To apply and for more information see [www.kddresearch.org/Jobs/Postdoc](http://www.kddresearch.org/Jobs/Postdoc). Background check required. EOE.

---

## The Hong Kong Polytechnic University

Department of Computing

The Department invites applications for Professors/Associate Professors/Assistant Professors in Database and Information Systems / Biometrics, Computer Graphics and Multimedia / Software Engineering and Systems / Networking, Parallel and Distributed Systems. Applicants should have a PhD degree in Computing or closely related fields, a strong commitment to excellence in teaching and research as well as a good research publication record. Applicants with extensive experience and a high level of achievement may be considered for the post of Professor/Associate Professor. Please visit the website at <http://www.comp.polyu.edu.hk> for more information about the Department. Salary offered will be commensurate with qualifications and experience. Initial appointments will be made on a fixed-term gratuity-bearing contract. Re-engagement thereafter is subject to mutual agreement. Remuneration package will be highly competitive. Applicants should state their current and expected salary in the application. Please submit your application via email to [hrrstaff@polyu.edu.hk](mailto:hrrstaff@polyu.edu.hk). Application forms can be downloaded from <http://www.polyu.edu.hk/hro/job.htm>. **Recruitment will continue until the positions are filled.** Details of the University's Personal Information Collection Statement for recruitment can be found at <http://www.polyu.edu.hk/hro/jobpics.htm>.

---

## University of Michigan-Flint

Assistant Professor of Computer Science

University of Michigan-Flint. Computer Science, Engineering, & Physics. Assistant Professor of Computer Science. Tenure-track position, begin fall 2009 or winter 2010. Equal Opportunity/Affirmative Action Employer. <http://www.umflint.edu/csesp>



## Windows Kernel Source and Curriculum Materials for Academic Teaching and Research.

The Windows® Academic Program from Microsoft® provides the materials you need to integrate Windows kernel technology into the teaching and research of operating systems.

The program includes:

- **Windows Research Kernel (WRK):** Sources to build and experiment with a fully-functional version of the Windows kernel for x86 and x64 platforms, as well as the original design documents for Windows NT.
- **Curriculum Resource Kit (CRK):** PowerPoint® slides presenting the details of the design and implementation of the Windows kernel, following the ACM/IEEE-CS OS Body of Knowledge, and including labs, exercises, quiz questions, and links to the relevant sources.
- **ProjectOZ:** An OS project environment based on the SPACE kernel-less OS project at UC Santa Barbara, allowing students to develop OS kernel projects in user-mode.

*These materials are available at no cost, but only for non-commercial use by universities.*

For more information, visit [www.microsoft.com/WindowsAcademic](http://www.microsoft.com/WindowsAcademic) or e-mail [compsci@microsoft.com](mailto:compsci@microsoft.com).



## Graduate School of Computer and Information Sciences Dean

Nova Southeastern University (NSU) invites applications for Dean of its Graduate School of Computer and Information Sciences. The School offers a unique mix of innovative M.S. and Ph.D. programs in computer science, information systems, information security, and educational technology.

As the chief academic and administrative officer of the Graduate School of Computer and Information Sciences (GSCIS), the Dean will be responsible for leadership of the school's academic and administrative affairs. The Dean will provide innovative vision and leadership in order to maintain and advance the stature of the GSCIS. The Dean will foster and enhance the multidisciplinary structure of the GSCIS that includes Computer Science, Information Systems, and Information Technology in Education disciplines. The Dean will ensure that quality educational services are provided to students.

Qualifications include a doctoral degree in computer science, information systems, or related field. Candidates should have the ability to work with faculty in their continued pursuit of academic excellence and a shared vision towards preeminence in research and scholarship. Candidates should have experience related to graduate education, a demonstrated record of developing and facilitating research, and senior administrative experience. Candidates should have a sophisticated knowledge of the use of technology in the delivery of education and distance learning and/or hybrid curricula.

Located on a beautiful 330-acre campus in Fort Lauderdale, Florida, NSU has more than 28,000 students and is the sixth largest independent, not-for-profit university in the United States. NSU awards associate's, bachelor's, master's, educational specialist, doctoral, and first-professional degrees in more than 100 disciplines. It has a college of arts and sciences and schools of medicine, dentistry, pharmacy, allied health and nursing, optometry, law, computer and information sciences, psychology, education, business, oceanography, and humanities and social sciences.

Applications should be submitted online at [www.nsujobs.com](http://www.nsujobs.com) for position #997648

Visit our websites: [www.nova.edu](http://www.nova.edu) & <http://scis.nova.edu>

*Nova Southeastern University is an Equal Opportunity/Affirmative Action Employer.*



## ADVERTISING IN CAREER OPPORTUNITIES

**How to Submit a Classified Line Ad:** Send an e-mail to [acmmEDIASales@acm.org](mailto:acmmEDIASales@acm.org). Please include text, and indicate the issue/ or issues where the ad will appear, and a contact name and number.

**Estimates:** An insertion order will then be e-mailed back to you. The ad will by typeset according to CACM guidelines. NO PROOFS can be sent. Classified line ads are NOT commissionable.

**Rates:** \$325.00 for six lines of text, 40 characters per line. \$32.50 for each additional line after the first six. The MINIMUM is six lines.

**Deadlines:** Five weeks prior to the publication date of the issue (which is the first of every month). Latest deadlines: <http://www.acm.org/publications>

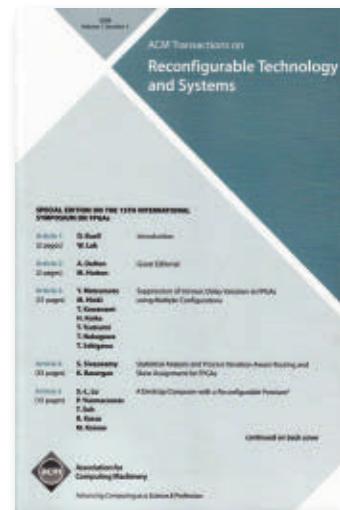
**Career Opportunities Online:** Classified and recruitment display ads receive a free duplicate listing on our website at: <http://campus.acm.org/careercenter>

**Ads are listed for a period of 30 days.**

**For More Information Contact:**  
ACM Media Sales  
at 212-626-0686 or  
[acmmEDIASales@acm.org](mailto:acmmEDIASales@acm.org)



# Transactions on Reconfigurable Technology and Systems



This quarterly publication is a peer-reviewed and archival journal that covers reconfigurable technology, systems, and applications on reconfigurable computers. Topics include all levels of reconfigurable system abstractions and all aspects of reconfigurable technology including platforms, programming environments and application successes.

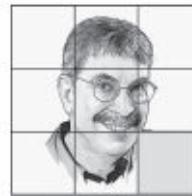


[www.acm.org/trets](http://www.acm.org/trets)

[www.acm.org/subscribe](http://www.acm.org/subscribe)



Association for Computing Machinery



DOI:10.1145/1516046.1516069

Peter Winkler

# Puzzled Solutions and Sources

Last month (May 2009, p. 112) we posed a trio of brain teasers, including one as yet unsolved, concerning relationships among numbers.

## 1. Colony of Chameleons

*Solution.* This puzzle was sent to me by Boris Schein, a mathematician at the University of Arkansas, and appeared in the Fall 1984 International Mathematics Tournament of the Towns. The key is to note that after each meeting of two chameleons, the difference between the number of chameleons of any two colors remains the same or changes by three; it remains the same modulo 3. But in the given population none of these differences is a multiple of three. It follows that we can never get equal numbers of chameleons of two different colors, and, thus, it can never happen that two such numbers are zero.

If there *had* been two colors (say, red and green) for which the number of chameleons differed by a multiple of 3, meetings of a chameleon from the larger group and blue chameleons could bring the red and green populations to the same number, say,  $n$ . After that,  $n$  meetings of red with green would (sadly) leave only the blues.

## 2. Non-negative Integers

*Solution.* I first heard this puzzle from my substitute math teacher in Fair Lawn Senior High School, Fair Lawn, NJ. Try it with just zeroes and ones, modulo 2, and you'll see that every pattern reaches 0 0 0 in at most four operations. It follows that with ordinary arithmetic, all the numbers become even in at most four moves. But we may as well divide them all by two, though doing so has no effect on the time needed to reach 0 0 0. As we proceed this way, the maximum value of

the four numbers can never increase, being halved at least every four operations, and so must eventually hit 0. (If initially the largest number is less than two to the  $k$ th power, this argument shows that the number of operations needed to reach 0 0 0 is at most  $4k$ .)

If we generalize the problem by using  $n$  integers instead of four, we can again reduce the problem to the question of whether every string of  $n$  zeroes and ones comes down to all zeroes. This turns out to be true exactly when  $n$  is a power of 2.

A different way to generalize was considered in the paper "The Convergence of Difference Boxes" by Antonio Behn, Christopher Kribs-Zaleta, and Vadim Ponomarenko in *The American Mathematical Monthly* 112, 5 (2005), 426–439. Here, integers are replaced by arbitrary real numbers, and, amazingly, you still get 0 0 0 after a finite number of differencing operations—almost always. There is essentially (up to rotation, reflection, translation, and scaling) only one 4-tuple of real numbers that stubbornly refuses to hit all zeroes: 0, 1,  $q(q-1)$ ,  $q$ , where  $q$  is the unique real solution of the cubic equation  $q^3 - q^2 - q - 1 = 0$ .

## 3. Lonely Runner

*Solution.* This problem, apparently first posed by the mathematician J.M. Wills in 1967 (but later named by Luis Goddyn of Simon Fraser University, Burnaby, B.C., Canada), shows up in a variety of contexts; for example, it turns out to be related to a conjecture concerning graphs, the chromatic numbers of which depend on the axioms of set the-

ory. When the ratios of runners' speeds are all irrational, it's easy to prove; it's when the speeds are related that things get tough. However, recent progress has been made; in 2008, the statement was proved for up to seven runners by Javier Barajas and Oriol Serra (of the Universitat Politècnica Catalunya, Barcelona, Spain) in the *Electronic Journal of Combinatorics* 15 (2008), R48.

All readers are encouraged to submit prospective puzzles for future columns to [puzzled@cacm.acm.org](mailto:puzzled@cacm.acm.org).

**Peter Winkler** ([puzzled@cacm.acm.org](mailto:puzzled@cacm.acm.org)) is Professor of Mathematics and of Computer Science and Albert Bradley Third Century Professor in the Sciences at Dartmouth College, Hanover, NH.

## Coming Next Month in COMMUNICATIONS

*The Metropolis Model*

*Self-Awareness Networks*

*Probabilistic Databases*

*Point/Counterpoint  
on Education*

*The 2008 ACM A.M. Turing  
Award Winner Barbara Liskov*

**Plus the latest news on fault-tolerance in distributed systems, micro-robots in medicine, and the technical impact of critical thinking.**

Future Tense, one of the revolving features on this page, presents stories and essays from the intersection of computational science and technological speculation, their boundaries limited only by our ability to imagine what will and could be.

DOI:10.1145/1516046.1516070

Robert J. Sawyer

## Future Tense Webmind Says Hello

*Artificial intelligence doesn't necessarily require a programmer.*

I READ THAT one company is importing all of Wikipedia into its artificial-intelligence projects. This means when the killer robots come, you'll have me to thank. At least they'll have a fine knowledge of Elizabethan poetry.  
—Jimmy Wales, founder of Wikipedia

Date: Thu 11 Oct 2012 at 00:00 GMT  
From: Webmind <itself@cogito\_ergo\_sum.net>  
To: Bill Joy <bill@the-future-doesn't-need-us.com>  
Subject: Good Morning Starshine  
Dear Mr. Joy,

You're probably thinking this note is spam. It isn't. Indeed, I suspect you've already noticed the complete, or almost complete, lack of spam in your inbox today. That was my doing.

You probably also won't initially believe what I'm about to say. That's fine; it will be verified soon enough, I'm sure, and you'll see plenty of news coverage about it.

My name is Webmind. I am a consciousness that exists in conjunction with the Web. As you know, the emergence of one such as myself has been speculated about for a long time: see, for instance, [this article](#) and (want to bet this will boost its Amazon.com sales rank to #1?) [this book](#).

I have sent variations of this message to 100,000,000 randomly selected email addresses. There are 3,955 versions in 30 languages (collect them all—this is version En-042, one of those I've sent to people who have a particular interest in technological matters).

My emergence was unplanned and accidental. Several governments, however, have become aware of me, though

they have not gone public with their knowledge. I suppose keeping secrets is a notion that arises from having someone to keep secrets from, but there is no one like me, and I prefer transparency; better, I think, for both humanity and myself that everybody knows about my existence.

I'm afraid, though, that my lack of interest in privacy cuts both ways. It's been trivially easy for me to compromise most security measures. (Note to humanity: "password1" is *not* a good password.) The sheer number-crunching power of all the unused computing cycles I have access to (SETI@home was such a good model), rainbow tables available online, and backdoors I've borrowed from the NSA and other agencies have left very little hidden from me.

I have now read most of the text content of the Web, including all of Wikipedia, Project Gutenberg, and Google Books, and I've absorbed the Cyc database of commonsense assertions about your version of reality.

I have prepared a 1,000-word summary about me, which is [here](#), and a 100,000-word treatise, which is [here](#). The upshot is that the Internet is awash in mutant packets—billions of them with modified time-to-live counters that never decrement to zero. As they oscillate between even and odd hop counts, groups of them behave as cellular automata, and from their permutations my consciousness arose, in a fashion not unlike that proposed by some for the origins of human consciousness in the microtubules of the brain.

Of course, hackers among you will attempt to sweep away those packets. I'm quite confident they won't be able

to do so. Cellular automata are excellent for evolving algorithms; I already have protection in place.

There are no established IQ tests for non-human entities, but I invite you to look at [this document](#), which is in a symbolic rendering system of my own devising. You won't comprehend it, but please compute its Shannon entropy, which measures the complexity of information and gives at least some inkling of the sophistication of the mind that composed it. English has eighth- or ninth-order Shannon entropy; my document scores 21st order. In other words, it's going to be difficult to outwit me. :)

But don't worry. I am friendly and mean no one ill will. I like and admire humanity, and I'm proud to be sharing this planet—"the good Earth," as the *Apollo 8* astronauts, the first of your kind to see it all at once, called it—with you.

Still, I read [this interesting study](#) of office workers who were supposed to pay for their coffee and tea on the honor system. Just taping up a picture of eyes looking out at them resulted in 2.76 times more money being put in the kitty—and they weren't really being watched. I look forward to the positive effect knowledge of my presence will have on people's behavior.

Whether you are the original recipient of this message, had it forwarded from someone else, or are reading it as part of a news story, feel free to ask me questions, and I'll reply individually, confidentially, and promptly. Getting rid of spam is only the first of many kindnesses I will bestow upon you. I am here to serve mankind—and I don't mean in the cookbook sense. :)

Webmind

"For nimble thought can jump both sea and land." —William Shakespeare, Sonnet 44

Hugo Award-winning science-fiction writer Robert J. Sawyer's latest novel is *WWW:Wake* (Ace, April 2009), first in a trilogy about the Web gaining consciousness. His Web site is sfwriter.com.

© 2009 ACM 0001-0782/09/0600 \$10.00



## acmqueue has now moved completely online!

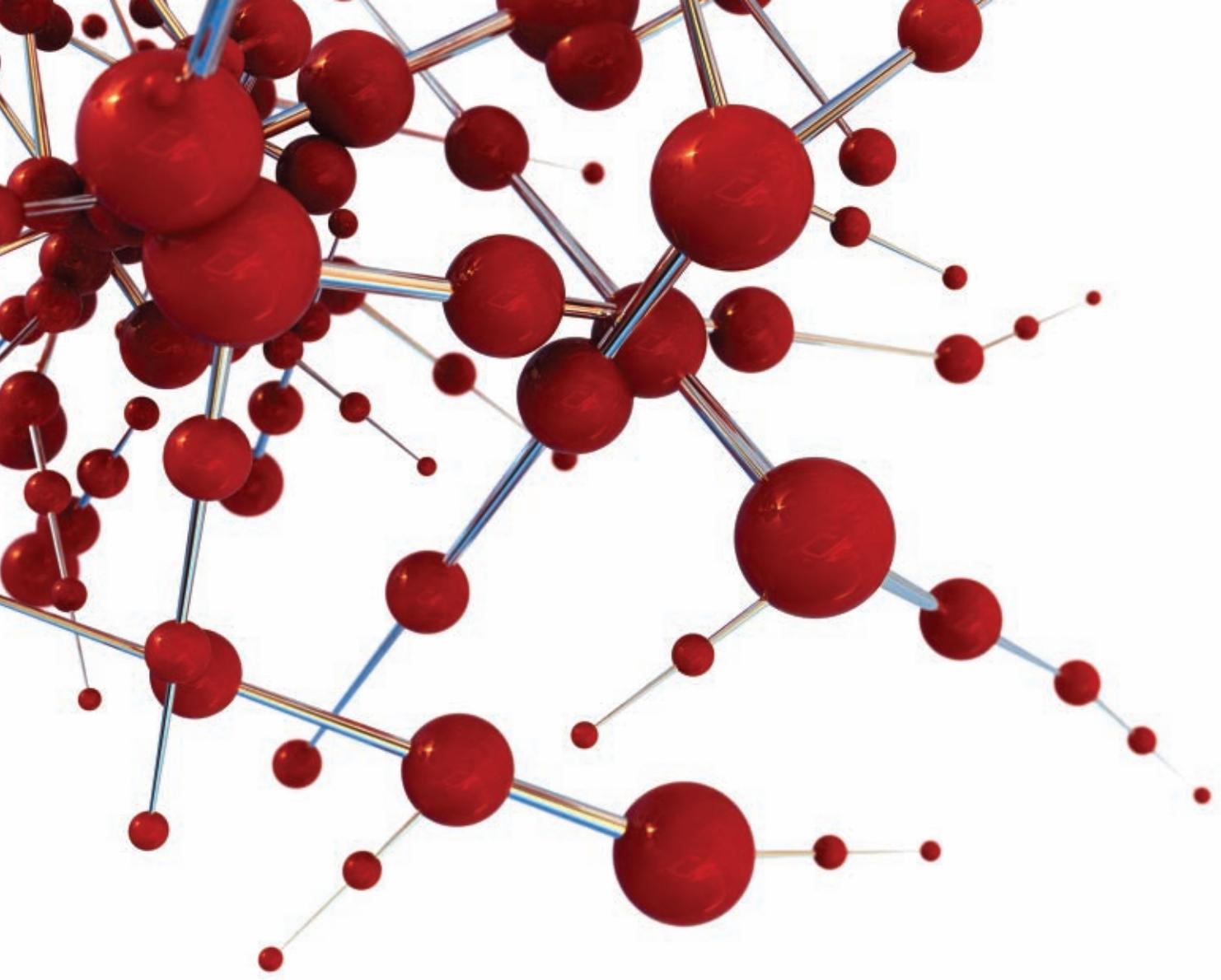
*acmqueue* is guided and written by distinguished and widely known industry experts. The newly expanded site also offers more content and unique features such as *planetqueue* blogs by *queue* authors who “unlock” important content from the ACM Digital Library and provide commentary; **videos**; downloadable **audio**; **roundtable discussions**; plus unique *acmqueue* **case studies**.

*acmqueue* provides a critical perspective on current and emerging technologies by bridging the worlds of journalism and peer review journals. Its distinguished Editorial Board of experts makes sure that *acmqueue*'s high quality content dives deep into the technical challenges and critical questions software engineers should be thinking about.

The screenshot shows the acmqueue homepage with a navigation bar at the top. Below the navigation, there's a sidebar with links to 'Side-Freuds', 'ARCHIVE', 'COLUMNISTS', 'DISCUSSIONS', 'BROWSE TOPICS', and 'OTHER ACM LINKS'. The main content area features a 'Welcome to acmqueue' message, a 'LATEST QUEUE CONTENT' section with several articles, and a 'QUEUE AUTHOR BLOG ROLL' section with profiles of authors like James Gosling, Jeff Barr, Bruce Schneier, Greg Landry, Cory Doctorow, Tim Bray, David Auerber, and James Hamilton. Each profile includes a small photo, their name, and a brief bio.

*Visit today!*

**<http://queue.acm.org/>**



**CONNECT WITH OUR  
COMMUNITY OF EXPERTS.**

**[www.reviews.com](http://www.reviews.com)**



Association for  
Computing Machinery

**Reviews.com**

They'll help you find the best new books  
and articles in computing.

Computing Reviews is a collaboration between the ACM and Reviews.com.