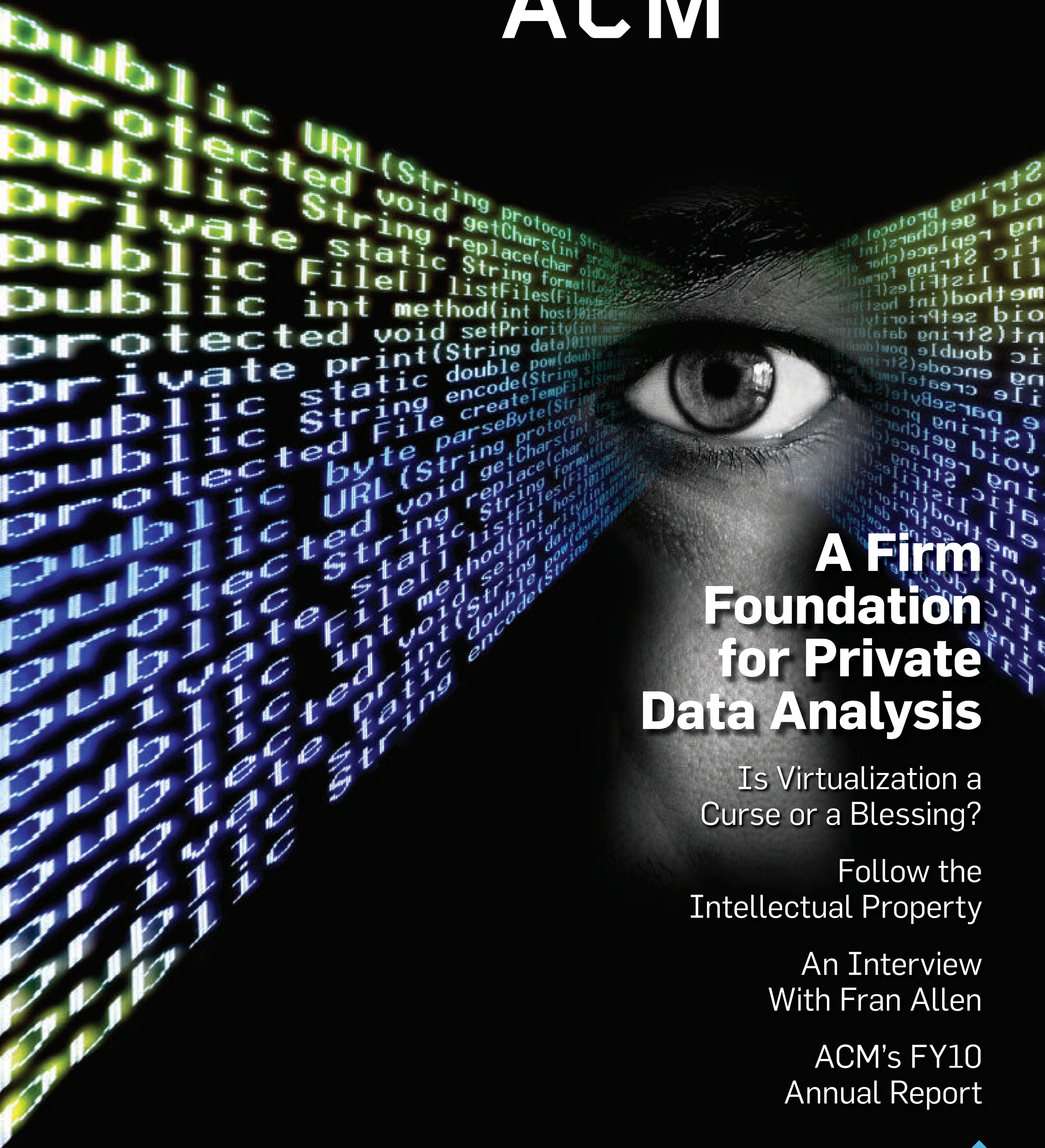
COMMUNICATIONS OF THE ACM

CACM.ACM.ORG

01/2011 VOL.54 NO.01



A Firm Foundation for Private Data Analysis

Is Virtualization a Curse or a Blessing?

Follow the Intellectual Property

An Interview With Fran Allen

ACM's FY10 Annual Report

Association for Computing Machinery

membership application & *digital library* order form

Association for Computing Machinery

Advancing Computing as a Science & Profession

Priority Code: AD10

You can join ACM in several easy ways:

Online	Phone	Fax
http://www.acm.org/join	*+1-800-342-6626 (US & Canada)* *+1-212-626-0500 (Global)*	*+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:

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

STUDENT MEMBERSHIP:

- o **ACM Student Membership: $19 USD**
- o **ACM Student Membership plus the ACM Digital Library: $42 USD**
- o **ACM Student Membership PLUS Print *CACM* Magazine: $42 USD**
- o **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

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
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.

o Visa/MasterCard　o American Express　o Check/money order

o Professional Member Dues ($99 or $198) $ ____________

o ACM Digital Library ($99) $ ____________

o Student Member Dues ($19, $42, or $62) $ ____________

Total Amount Due $ ____________

Card # | Expiration date

Signature

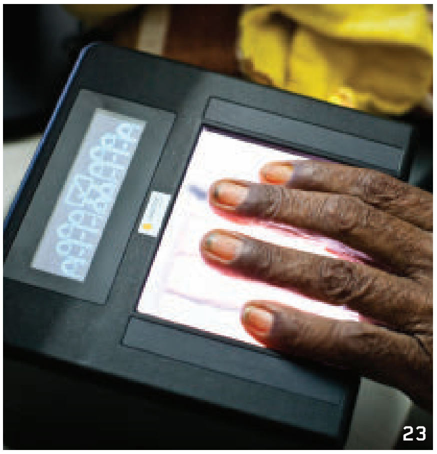
COMMUNICATIONS OF THE ACM

Departments

Last Byte

Association for Computing Machinery
Advancing Computing as a Science & Profession

News



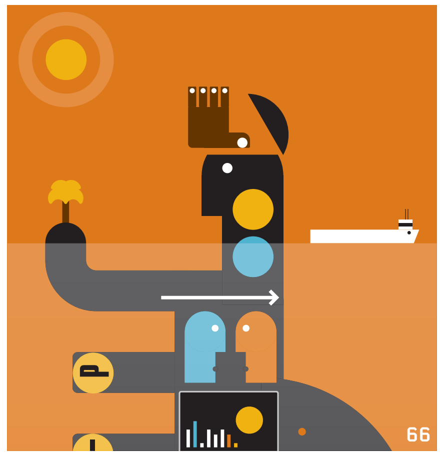
Viewpoints

PHOTOGRAPH BY SANJIT DAS/PANOS

Practice

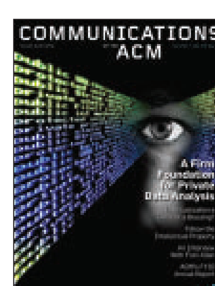
Articles' development led by acmqueue
queue.acm.org

Contributed Articles



Review Articles

Research Highlights



About the Cover: Preserving privacy in an online world remains one of the industry's most exhaustive challenges. While great progress has been made, vulnerabilities indeed remain. This month's cover story by Cynthia Dwork (p. 86) spotlights the difficulties involved in protecting statistical databases, where the value of accurate statistics about a set of respondents often compromises the privacy of the individual.

ILLUSTRATION BY PETER GRUNDY

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

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 U.S. Public Policy Office
Cameron Wilson, Director
1828 L Street, N.W., Suite 800
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 (541) 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

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

Subscriptions
An annual subscription cost is included in ACM member dues of $99 ($40 of which is allocated to a subscription to *Communications)*; for students, cost is included in $42 dues ($20 of which is allocated to a *Communications* subscription). A nonmember annual subscription is $100.

Single Copies
Single copies of *Communications of the ACM* are available for purchase. Please contact 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:10.1145/1866739.1866740

Moshe Y. Vardi

Where Have All the Workshops Gone?

My initiation into the computing-research community was a workshop on "Logic and Databases" in 1979. I was the only graduate student attending that workshop;

my graduate advisor was invited, and he got permission from the organizers to bring me along. In spite of the informality of the event I was quite in awe of the senior researchers who attended the workshop. In fact, I was quite in shock when one of them, an author of a well-respected logic textbook, proved to be far from an expert in the subject matter of his book.

Throughout the 1980s, workshops continued to be informal gatherings of researchers mixing networking with work-in-progress presentations and intellectually stimulating discussions. A workshop was typically a rather intimate gathering of specialists; an opportunity to invite one's scientific friends to get together. While conferences were the place to present polished technical results, workshops were a place to see if your colleagues were as impressed with your new results or directions as you were. The pace was leisurely, many presentations were done on blackboards, and it was perfectly acceptable to ask questions during presentations. Organizers may have posted an occasional "call for abstracts," but never a "call for papers." In fact, workshops typically had no formal proceedings.

Such informal workshops are almost extinct today. As selective conferences become our dominant way of publishing, workshops have gradually become mini-conferences. Today's workshops have typically large program committees, calls for papers, deadlines, and all the other accoutrements of computing-research conferences. What they usually lack is the prestige of major conferences. Furthermore, most workshops today do publish proceedings, before or after the meeting, which means a workshop paper cannot be resubmitted to a conference. As a result, today's workshops do not attract papers of the same quality as those submitted to major conferences.

Workshops have become, I am afraid to say, simply second-rate conferences. Yes, I am sure there are exceptions to this, but I believe my description does apply to the vast majority of today's computing-research workshops. It is not uncommon to see workshops where the size of the program committee exceeds the number of papers submitted to the workshop. It is not uncommon to see deadlines extended in the hope of attracting a few more submissions.

I miss the old workshops. Regardless of what one thinks of computing-research conferences (our community is now engaged in serious discussions on the advantages and disadvantages of these meetings), informal workshops played an important role in the computing-research ecosystem. Many preliminary results improved significantly as a result of feedback received from discussions carried out during these gatherings. The disappearance of such workshops is, in my opinion, a loss to our community.

I am a big fan of Schloss Dagstuhl, a workshop facility near the small town of Wadern in Germany. Schloss Dagstuhl was built as a manor house of a German prince in 1760. It was converted into the International Conference and Research Center for Computer Science in 1989, now called Leibniz Center for Informatics. The first week-long seminar (Dagstuhl workshops are called seminars) took place in August 1990. Since then, Dagstuhl has hosted close to 800 seminars, drawing about 30,000 participants. In addition to week-long seminars, Dagstuhl hosts perspectives workshops, summer schools, retreat stays of research guests, and the like. If you receive an invitation to a Dagstuhl seminar, accept it! The facility offers a good library and an outstanding wine cellar. The rural location facilitates both group and one-on-one interactions. In a nutshell, Dagstuhl is the place to experience the tradition of workshops as informal scientific gatherings. Its contributions to computing research over the past 20 years are incalculable. It is no wonder that the National Institute of Informatics in Japan recently created a similar center in Shonan, near Tokyo.

This brings me to a question that has been bothering me for years. Call it "Dagstuhl Envy," but why don't we have a North American "Dagstuhl"? There are several facilities in North America to host mathematics workshops, for example, the Banff International Research Station, and these are often used for workshops on topics in theoretical computer science. There is, however, no facility dedicated for general computing-research workshops. It would probably take about $10 million to build such a facility and approximately $2 million–$3 million annually to cover operating costs. These are modest sums in the context of the size of the North American computing-research portfolio and the size of the North American information-technology industry. Can we make it happen?

Moshe Y. Vardi, EDITOR-IN-CHIEF

DOI:10.1145/1866739.1866741

To Change the World, Take a Chance

Some of what Constantine Dovrolis said in the Point/Counterpoint "Future Internet Architecture: Clean-Slate Versus Evolutionary Research" (Sept. 2010) concerning an evolutionary approach to developing Internet architecture made sense, and, like Jennifer Rexford on the other side, I applaud and encourage the related "evolutionary" research. But I found his "pragmatic vision" argument neither pragmatic nor visionary. Worse was the impudence of the claim of "practicality."

Mid-20th century mathematician Morris Kline said it best when referring to the history of mathematics: "The lesson of history is that our firmest convictions are not to be asserted dogmatically; in fact they should be most suspect; they mark not our conquests but our limitations and our bounds."

For example, it took 2,000 years for geometry to move beyond the "pragmatism" of the parallel postulate, some 200 years for Einstein to overtake Newton, 1,400 years for Copernicus to see beyond Ptolemy, and 10,000 years for industrialization to supplant agriculture as the dominant economic activity. The Internet's paltry 40–50-year history is negligible compared to these other clean-slate revolutions.

Though such revolutions generally fail, failure is often the wellspring of innovation. Honor and embrace it. Don't chide it as "impractical." The only practical thing to do with this or any other research agenda is to open-mindedly test our convictions and assumptions over and over...including any clean-slate options.

I worry about the blind spot in our culture, frequently choosing "practical effort" over bolder investment, to significantly change things. Who takes the 10,000-, 1,000-, or even 100-year view when setting a research agenda? Far too few. Though "newformers" fail more often than the "practical" among us, they are indeed the ones who change the world.

CJ Fearnley, Upper Darby, PA

What Deeper Implications for Offshoring?

As someone who has known offshoring for years, I was drawn to the article "How Offshoring Affects IT Workers" by Prasanna B. Tambe and Lorin M. Hitt (Oct. 2010) but disappointed to find a survey-type analysis that essentially confirmed less than what most of us in the field already know. For example, at least one reason higher-salaried workers are less likely to be offshored is they already appreciate the value of being able to bridge the skill and cultural gap created by employing offshore workers.

I was also disappointed by the article's U.S.-centric view (implied at the top in the word "offshoring"). What about how offshoring affects IT workers in countries other than the U.S.? In my experience, they are likewise affected; for example, in India IT workers are in the midst of a dramatic cultural upheaval involving a high rate of turnover.

While seeking deeper insight into offshoring, I would like to ask someone to explain the implications of giving the keys to a mission-critical system to someone in another country not subject to U.S. law? Imagine if the relationships between countries would deteriorate, and the other country would seize critical information assets? We have pursued offshoring for years, but I have still not heard substantive answers to these questions.

Mark Wiman, Atlanta, GA

Authors' Response:

With so little hard data on outsourcing, it is important to first confirm some of the many anecdotes now circulating. The main point of the article was that the vulnerability of occupations to offshoring can be captured by their skill sets and that the skills story is not the only narrative in the outsourcing debate.

The study was U.S.-centric by design. How offshoring affects IT workers in other countries is important, but the effects of offshoring on the U.S. IT labor market merits its own discussion.

Misappropriation of information has been studied in the broader outsourcing context; see, for example, Eric K. Clemons's and Lorin M. Hitt's "Poaching and the Misappropriation of Information" in the Journal of Management Information Systems *21, 2 (2004), 87–107.*

Prasanna B. Tambe, New York, NY
Lorin M. Hitt, Philadelphia, PA

Interpreting Data 100 Years On

Looking to preserve data for a century or more involves two challenging orthogonal problems, one—how to preserve the bits—addressed by David S.H. Rosenthal in his article "Keeping Bits Safe: How Hard Can It Be?" (Nov. 2010). The other is how to read and interpret them 100 years on when everything might have changed—formats, protocols, architecture, storage system, operating system, and more. Consider the dramatic changes over just the past 20 years. There is also the challenge of how to design, build, and test complete systems, trying to anticipate how they will be used in 100 years. The common, expensive solution is to migrate all the data every time something changes while controlling costs by limiting the amount of data that must be preserved in light of deduplication, legal obsolescence, librarians, archivists, and other factors.

For more on data interpretation see:

1. Lorie, R.A. A methodology and system for preserving digital data. In *Proceedings of the Joint Conference on Digital Libraries* (Portland, OR, July 2002), 312–319.

2. Lorie, R.A. Long-term preservation of digital information. In *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries* (Roanoke, VA, Jan. 2001), 346–352.

3. Rothenberg, J. *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation.* Council on Library & Information Resources, 1999.

Robin Williams, San Jose, CA

Author's Response:

As Williams says, the topic of my article was not interpreting preserved bits. Jeff Rothenberg drew attention to the threat of format obsolescence in Scientific American *(Jan. 1995), a focus that has dominated digital preservation ever since. Rothenberg was writing before the Gutenberg-like impact of the Web transformed digital formats from private, to applications, to publishing medium, leading him and others to greatly overestimate the obsolescence risk of current formats.*

I am unable to identify a single format widely used in 1995 that has since become obsolete but would welcome an example. At the 2010 iPres conference (http://www.ifs.tuwien.ac.at/dp/ipres2010/) I asked the audience whether any of them had ever been forced to migrate the format of preserved content to maintain renderability, and no one had.

Format obsolescence is clearly a threat that must be considered, but compared to the technical and economic threats facing the bits we generate, it is insignificant and the resources devoted to it are disproportionate.

David S.H. Rosenthal, Palo Alto, CA

Objects Always! Well, Almost Always

Unlike Mordechai Ben-Ari's Viewpoint "Objects Never, Well, Hardly Ever!" (Sept. 2010), for me learning OOP was exciting when I was an undergraduate almost 30 years ago. I realized that programming is really a modeling exercise and the best models reduce the communication gap between computer and customer. OOP provides more tools and techniques for building good models than any other programming paradigm.

Viewing OOP from a modeling perspective makes me question Ben-Ari's choice of examples. Why would anyone expect the example of a car to be applicable to a real-time control system in a car? The same applies to the "interface" problem in supplying brake systems to two different customers. There would then be no need to change the "interface" to the internal control systems, contrary to Ben-Ari's position.

Consider, too, quicksort as implemented in Ruby:

```
def quicksort(v)
    return v if v.nil? or
    v.length <= 1
    less, more = v[1..-1].
    partition { |i| i < v[0] }
    quicksort(less) + [v[0]] +
    quicksort(more)
end
```

This concise implementation shows quicksort's intent beautifully. Can a nicer solution be developed in a non-OOP language? Perhaps, but only in a functional one. Also interesting is to compare this solution with those in 30+ other languages at http://en.wikibooks.org/wiki/Algorithm_implementation/Sorting/Quicksort, especially the Java versions. OO languages are not all created equal.

But is OOP dominant? I disagree with Ben-Ari's assertion that "...the extensive use of languages that support OOP proves nothing." Without OOP in our toolbox, our models would not be as beautiful as they could be. Consider again Ruby quicksort, with no obvious classes or inheritance, yet the objects themselves—arrays, iterators, and integers—are all class-based and have inheritance. Even if OOP is needed only occasionally, the fact that it is needed at all and subsumes other popular paradigms (such as structured programming) supports the idea that OOP is dominant.

I recognize how students taught flowcharts first (as I was) would have difficulty switching to an OO paradigm. But what if they were taught modeling first? Would OOP come more naturally, as it did for me? Moreover, do students encounter difficulties due to the choice of language in their first-year CS courses? I'm much more comfortable with Ruby than with Java and suspect it would be a better introductory CS language. As it did in the example, Ruby provides better support for the modeling process.

Henry Baragar, Toronto

I respect Mordechai Ben-Ari's Viewpoint (Sept. 2010), agreeing there is neither a "most successful" way of structuring software nor even a "dominant" way. I also agree that research into success and failure would inform the argument. However, he seemed to have fallen into the same all-or-nothing trap that often permeates this debate. OO offers a higher level of encapsulation than non-OO languages and allows programmers to view software realistically from a domain-oriented perspective, as opposed to a solution/machine-oriented perspective.

The notion of higher levels of encapsulation has indeed permeated many aspects of programmer thinking; for example, mobile-device and Web-application-development frameworks leverage these ideas, and the core tenets of OO were envisioned to solve problems involving software development prevalent at that time.

Helping my students become competent, proficient software developers, I find the ones in my introductory class move more easily from OOP-centric view to procedural view than in the opposite direction, but both types of experience are necessary, along with others (such as scripting). So, for me, how to start them off and what to emphasize are important questions. I like objects-first, domain-realistic software models, moving as needed into the nitty-gritty (such as embedded network protocols and bus signals). Today's OO languages may indeed reflect deficiencies, but returning to an environment with less encapsulation would mean throwing out the baby with the bathwater.

James B. Fenwick Jr., Boone, NC

The bells rang out as I read Mordechai Ben-Ari's Viewpoint (Sept. 2010)—the rare, good kind, signaling I might be reading something of lasting importance. In particular, his example of an interpreter being "nicer" as a case/switch statement; some software is simply action-oriented and does not fit the object paradigm.

His secondary conclusion—that Eastern societies place greater emphasis on "balance" than their Western counterparts, to the detriment of the West—is equally important in software. Objects certainly have their place but should not be advocated to excess.

Alex Simonelis, Montreal

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.

DOI:10.1145/1866739.1866742

In the Virtual Extension

To ensure the timely publication of articles, Communications *created the Virtual Extension (VE) to expand the page limitations of the print edition by bringing readers the same high-quality articles in an online-only format. VE articles undergo the same rigorous review process as those in the print edition and are accepted for publication on merit. The following synopses are from articles now available in their entirety to ACM members via the Digital Library.*

viewpoint

DOI: 10.1145/1866739.1866764

The Ephemeral Legion: Producing an Expert Cyber-Security Work Force from Thin Air

Michael E. Locasto, Anup K. Ghosh, Sushil Jajodia, and Angelos Stavrou

Although recent hiring forecasts (some *thousands* of new cyber-security professionals over the next three years) by both the NSA and DHS show a strong demand for cyber-security skills, such a hiring spree seems ambitious, to say the least. The current rate of production of skilled cyber-security workers satisfies the appetite of neither the public nor private sector, and if a concerted effort to drastically increase this work force is not made the U.S. will export high-paying information security jobs. In a global economy, such a situation isn't necessarily a bad outcome, but it poses several challenges to the U.S.'s stated cyber-security plans.

The authors believe the creation of a significant cyber-security work force is not only feasible, but also will help ensure the economic strength of the U.S. Beyond offering immediate economic stimulus, the nature of these jobs demands they remain in the U.S. for the long term, and they would directly support efforts to introduce information technology into the health care and energy systems in a secure and reliable fashion. Without a commitment to *educating* such a work force, it is impossible to *hire* such a work force into existence.

From the authors' point of view, far too few workers are adequately trained mostly because traditional educational mechanisms lack the resources to *effectively* train large numbers of experienced, knowledgeable cyber-security specialists. Just as importantly, many of the current commercial training programs and certifications focus on teaching skills useful for fighting the last cyberwar, not the current, nor future ones.

contributed article

DOI: 10.1145/1866739.1866766

On the Move, Wirelessly Connected to the World

Peter Fröhlich, Antti Oulasvirta, Matthias Baldauf, and Antti Nurminen

Is it possible to experience real-world landmarks through a wave, gaze, location coordinates, or touch, prompting delivery of useful digital information? Today's mobile handheld devices offer opportunities never before possible for interacting with digital information that responds to users' physical locations. But mobile interfaces have only limited input capabilities, usually just a keyboard and audio, while emerging multimodal interaction paradigms are beginning to take advantage of user movements and gestures through sensors, actuators, and content. For example, tourists asking about an unfamiliar landmark might point at it intuitively and would certainly welcome a handheld computer that responds directly to that interest. When passersby provide directions, the description might include local features, as in, say, "Turn right after the red building and enter through the metal gates." They, too, would welcome being able to see these features represented in a directly recognizable way on their handhelds. Or when following a route to a remote destination, they would want to know the turns and distances they would need to take through tactile or auditory cues, without having to switch their gaze between the environment and the display.

This article explores the synthesis of several emerging research trends called Mobile Spatial Interaction, or MSI (http://msi.ftw.at), covering new interaction techniques that let users interact with physical, natural, and urban surroundings through today's sensor-rich mobile devices.

contributed article

DOI: 10.1145/1866739.1866765

OpenSocial: An Enabler for Social Applications on the Web

Matthias Häsel

Social networking and open interfaces can be seen as representative of two characteristic trends to have emerged in the Web 2.0 era, both of which have evolved in recent years largely independently of each other. A significant portion of our social interaction now takes place on social networks, and URL-addressable APIs have become an integral part of the Web. The arrival of OpenSocial heralds a new standard uniting these two trends by defining a set of programming interfaces for developing social applications that are interoperable on different social network sites.

OpenSocial applications are interoperable within the context of multiple networks and build on standard technologies such as HTML and JavaScript. The advent of OpenSocial increases a developer's scope and productivity considerably, as it means that applications need only be developed once, and can then be implemented within the context of any given container that supports the standard. Meanwhile, operators of social network sites are presented with the opportunity to expand on their own existing functionalities with a host of additional third-party applications, without having to relinquish control over their user data in the process.

Until it was made public in November 2007, the OpenSocial standard was driven primarily by Google. The standard was not suited to productive use at that time however, as there were several shortcomings with respect to the user interface and security. The specification is now managed by the non-profit OpenSocial Foundation and, with its 0.8 version, a stable state suitable for commercial use has been reached.

By discovering, welcoming, and nurturing talent from all corners of the computing arena, ACM can truly be distinguished as the *world's* leading computing society.

DOI:10.1145/1866739.1866768 Wendy Hall

ACM's Annual Report

It has truly been a banner year for ACM. We firmly established ACM hubs in Europe, India, and China after years of exhaustive efforts to expand the Association's global reach. We moved ACM's commitment to women in computing to a new level with further development of the ACM Women's Council and the launch of ACM-W activities in India. And, (dare I say, not surprisingly), ACM membership ended the year at another all-time high.

Increasing ACM's relevance and influence in the global computing community has been a top priority throughout my presidency. By sharing ACM's array of valued resources and services with a borderless audience, and by discovering, welcoming, and nurturing talent from all corners of the computing arena, ACM can truly be distinguished as the *world's* leading computing society. It was therefore a great honor to host the opening days of ACM Europe, ACM India, and ACM China. The global stage has indeed been set for ACM to flourish internationally as never before.

ACM continues to play a leadership role in improving the image and health of the computing discipline. This is particularly evident with the Association's work in influencing change for women pursuing a career in computing. Through committees and initiatives such as ACM Women's Council, The Coalition to Diversify Computing, and the Computer Science Teachers Association (CSTA), ACM is helping to build balance, diversity, and opportunity for all who may be interested in technology. It was particularly inspiring to see members of ACM-W on hand at the launching of ACM India earlier this year, encouraging their counterparts to join forces to improve working conditions for women in computing in India. The Association's commitment to addressing the challenges faced by women in the field today is one that every member should applaud.

The fact that membership has continued to increase for eight consecutive years is testament to the ever-growing awareness of ACM's commitment to supporting the professional growth of its members. Indeed, by the end of FY10—spanning an acutely challenging year in global economies—the Association's membership stood at an all-time high, thus cementing ACM's position as the largest educational and scientific computing society in the world.

The following pages summarize some of the highlights of a busy year in the life of ACM. While much has been accomplished, there is still much to be done. In FY11, the Association will continue to grow initiatives in India, China, and Europe as well as identify other regions of the world where it is feasible for ACM to increase its level of activity. Improving the image and health of our discipline and field requires the concerted commitment of every ACM volunteer, board, chapter, committee, and member. It is through the support of devoted volunteers, members, and industry partners that ACM is able to make a real difference in the future of computing. It has been a pleasure to serve as your president during a time of such great promise.

Wendy Hall, ACM PRESIDENT

ACM's Annual Report for FY10

ACM, the Association for Computing Machinery, is an international scientific and educational organization dedicated to advancing the arts, sciences, and applications of information technology.

Publications

The centerpiece of the ACM Publication portfolio is the ACM Digital Library. During the past year, 21,000 full-text articles were added to the DL, bringing total holdings to 281,000 articles. ACM's *Guide to Computing Literature* is an integral part of the DL, providing an increasingly comprehensive index to the literature of computing. More than 230,000 works were added to the bibliographic database in FY10, bringing the total *Guide* coverage to over 1.52 million works.

Significant enhancements were made to the Digital Library and *Guide* this year, including a major reorganization of the core citation pages and to ACM bibliometrics. Along with content reformation, there is now greater ease of navigation and a greater selection of tools and resources.

ACM currently publishes 40 journals and *Transactions*, 10 magazines, and 23 newsletters. In addition, it provides primary online distribution for 10 periodicals through the Digital Library. During FY10, ACM added 364 conference and related workshop proceedings to the DL, including 45 in ACM's *International Conference Proceedings Series*.

Two ACM magazines were relaunched during FY10. *Crossroads*, the ACM student magazine became *XRDS*, with a more expansive editorial scope and a more modern look to appeal to the student audience. *ACM Inroads* was transformed from the *SIGCSE Bulletin* newsletter to an ACM magazine with a wider variety of content for computer science educators.

Periodicals that were approved by the Publications Board and are now on the launching pad for FY11: *ACM Transactions on Management Information Systems*; *ACM Transactions on Intelligent Systems and Technology*; and *ACM Transactions on Interactive Intelligent Systems*.

Education

ACM continues to work with multiple organizations on important issues related to the image of computing and the health of the discipline and profession. In the second year of an NSF grant to develop a more relevant image for computing, ACM worked in tandem with WGBH-Boston in the creation of a new messaging campaign called "Dot Diva." The campaign, which rolled out in the U.S. last month, is focused on ways to engage young girls with the potential of computing.

ACM and the Association for Information Systems (AIS) jointly developed new curriculum guidelines for undergraduate degree programs in information systems that for the first time include both core and elective courses suited to specific career tracks. Released in May, IS 2010 is aimed at educating graduates who are prepared to enter the work force equipped with IS-specific as well as foundational knowledge and skills. The report describes the seven core courses that must be covered in every IS program and the curriculum can be adapted for schools of business, public administration, and information science or informatics.

ACM's Computer Science Teachers Association (CSTA) continues to support and promote the teaching of computer science at the K–12 level as well as providing opportunities and resources for teachers and students to improve their understanding of computing disciplines. CSTA's mission is to ensure computer science emerges as a viable discipline in high schools and middle schools; it is a key partner in ACM's effort to see real computer science count at the high school level.

Professional Development

The Professional Development Committee spearheaded the development of a new product for practitioners and managers this year called Tech Packs. These integrated learning packages were created to provide a resource for emerging areas of computing designed around an annotated bibliography of resources selected from ACM's Digital Library, ACM's online book and course offerings, and non-ACM resources created by experts' recommendations on current computing topics. A Tech Pack comprises a set of fundamentally important articles on a subject with new material to provide a context and perspective on the theme. The goal is that communities might be built around Tech Packs with members commenting on selected resources and suggesting new ones.

The Professions Board Case Study program took off this year, with the first of several planned studies available online and in print. The program was designed to take an in-depth look at a company or product or technology from its inception to future plans by interviewing some of the key players involved. The inaugural case study was posted on the *ACM Queue* site and published in *Communications of the ACM*. The article was quickly slashdotted, and drew over 50,000 unique visits to the *Queue* site by the end of the fiscal year.

Traffic to the *Queue* Web site (http://queue.acm.org/) more than doubled this year over last. By the end of FY10, the site delivered nearly a million page views to nearly half a million readers.

Public Policy

Members of the U.S. Public Policy Council of ACM (USACM) had an active year interacting with policymakers in areas of e-voting, privacy, and security, as well as testifying before Congressional committees and helping develop principles for increasing the usability of government information online. Among the issues tackled this year, USACM joined a task force for the Future of American Innovation urging more funding for basic research and STEM education. Members also expressed concerns with the Cybersecurity Act of 2009, provided constructive comments on a draft of the Internet Privacy bill, and issued a response to e-voting legislation and Internet voting as it relates to military and overseas voters.

The ACM Committee on Computers and Public Policy aids the Association with respect to a variety of internationally relevant issues pertaining to computers and public policy. The online ACM Forum on Risks to the Public in Computer and Related Systems and the "Inside Risks" column published in *Communications of the ACM* reflect CCPP's long-standing dedication to policy issues on a global scale.

ACM played an active role in the National Center for Women and Information Technology (NCWIT) this year, particularly with regard to the K–12 Alliance—a coalition of educational organizations interested in helping young girls develop an interest in computer science and information technology.

The ACM Education Policy Committee (ACM EPC), established to educate policymakers about the appropriate role of computer science in the K–12 system, made major progress in bringing computer science into STEM discussions at all levels of government. Through the work of EPC, computer science is now explicitly recognized in key federal legislation as well as Department of Education regulations and initiatives. Indeed, EPC successfully led an effort that resulted in the U.S. House of Representatives declaring the week of December 7th as National Computer Science Education Week. ACM took a leadership role in steering the first CSEDWeek (held Dec. 6–12, 2009); a role the organization reprised for the second CSEDWeek held last month.

Students

ACM's renowned International Collegiate Programming Contest (ICPC), sponsored by IBM, drew 22,000 contestants representing 1,931 universities from 82 countries. The finals were held in Harbin, China, where 103 teams competed. The top four teams won gold medals as well as employment or internship offers from IBM.

Last January, *ACM Queue*'s Web site offered an online programming competition based on the ICPC. The inaugural *Queue* ICPC Challenge—open to all *Queue* readers (not just students)—was a huge success.

The ACM Student Research Competition (SRC), sponsored by Microsoft Research, provides a unique forum for undergraduate and graduate students to present their original research at well-known ACM-sponsored and co-sponsored conferences before a panel of judges and attendees. This venue draws an increasing number of students each year as it affords an exceptional opportunity for students to showcase their work and develop their skills as researchers.

ACM continues to cultivate its partnerships with leading technology companies, including Microsoft and Computer Associates, to offer valuable tools specifically for ACM student members. Available under the Student Academic Initiative is the Microsoft Developer Academic Alliance now offering student members free and unlimited access to over 100 software packages and the CA Academic Initiative including access to complimentary CA software.

ACM-W's Scholarship program, which offers stipends to select students to attend research conferences worldwide, was given an extra financial boost this year with new funding from the Bangalore-based global IT services corporation Wipro and Sun Microsystems (prior to the Oracle takeover). The increased funding will allow ACM-W to offer students larger scholarships as well as enable participation by women in both international and local events.

International

ACM Europe and ACM India were launched in FY10. Both organizations operate with councils established around three subcommittees: chapters; conferences; and members, awards, and volunteer leaders with the goal of increasing the presence of and generating interest in these popular ACM services.

The number of ACM Fellows, Distinguished, and Senior members from Europe has increased as has the number of ACM chapters throughout Europe.

Moreover, Microsoft Research Europe provided $50,000 to enhance the ACM Distinguished Speakers Program with a goal of delivering more high-quality, ACM-branded lectures

ACM Council

PRESIDENT
Wendy Hall

VICE PRESIDENT
Alain Chesnais

SECRETARY/TREASURER
Barbara Ryder

PAST PRESIDENT
Stuart I. Feldman

SIG GOVERNING BOARD CHAIR
Alexander Wolf

PUBLICATIONS BOARD CO-CHAIRS
Ronald Boisvert, Holly Rushmeier

MEMBERS-AT-LARGE
Carlo Ghezzi, Anthony Joseph, Mathai Joseph, Kelly Lyons, Bruce Maggs, Mary Lou Soffa, Fei-Yue Wang

SGB COUNCIL REPRESENTATIVES
Joseph A. Konstan, Robert A. Walker, Jack Davidson

ACM Headquarters

EXECUTIVE DIRECTOR/CEO
John R. White

DEPUTY EXECUTIVE DIRECTOR/COO
Patricia M. Ryan

2009 ACM Award Recipients

A.M. TURING AWARD
Charles P. Thacker

ACM-INFOSYS FOUNDATION AWARD IN THE COMPUTING SCIENCES
Eric Brewer

ACM/AAAI ALLEN NEWELL AWARD
Michael I. Jordan

THE 2009–2010 ACM-W ATHENA LECTURER AWARD
Mary Jane Irwin

GRACE MURRAY HOPPER AWARD
Tim Roughgarden

ACM-IEEE CS 2010 ECKERT-MAUCHLY AWARD
William J. Dally

KARL V. KARLSTROM OUTSTANDING EDUCATOR AWARD
Matthias Felleisen

OUTSTANDING CONTRIBUTION TO ACM AWARD
Moshe Y. Vardi

DISTINGUISHED SERVICE AWARD
Edward Lazowska

PARIS KANELLAKIS THEORY AND PRACTICE AWARD
Mihir Bellare and Phillip Rogaway

SOFTWARE SYSTEM AWARD
VMware Workstation 1.0, Mendel Rosenblum, Edouard Bugnion, Scott Devine, Jeremy Sugerman, Edward Wang

EUGENE L. LAWLER AWARD AND INFORMATICS
Gregory D. Abowd

ACM-IEEE KEN KENNEDY AWARD
Francine Berman

DOCTORAL DISSERTATION AWARD
Craig Gentry

ACM PRESIDENTIAL AWARD
Mathai Joseph, Elaine J. Weyuker

HONORABLE MENTION
Haryadi S. Gunawi, Andre Platzer, Keith Noah Snavely

in Europe.

Through the efforts of ACM India, launched last January in Bangalore, the number of chapters in India has more than doubled over the last 12 months and professional membership is up over 50%.

ACM-W hosted a "Women in Computing" event at the ACM India festivities to encourage India's women in computing to network and organize to form a community that works toward improving working and learning environments for all women in computing in India.

A new ACM China Council was officially launched in Beijing in June. Established to recognize and support ACM members and activities in China, the new council comprises a cross section of the computer science and information technology community committed to increasing the visibility and relevance of ACM in China. The group is also exploring several cooperative efforts with the China Computer Federation (CCF).

The Publications Board is playing a significant role in ACM's China Initiative. The Board approved three specific steps aimed at improving the exposure of Western audiences to Chinese research and to further improve ties between ACM and the CCF by including Chinese translations of articles from *Communications of the ACM* in the Digital Library; hosting two CCF journals in the Digital Library; and developing a co-branded ACM/CCF journal.

The first two SIGSPATIAL chapters in China and Australia were chartered this year.

Balance Sheet: June 30, 2010 (in Thousands)

ASSETS	
Cash and cash equivalents	$26,463
Investments	51,720
Accounts receivable and other assets	6,793
Deferred conference expenses	5,151
Fixed assets, net of accumulated depreciation and amortization	781
Total Assets	**$90,908**
LIABILITIES AND NET ASSETS	
Liabilities:	
Accounts payable, accrued expenses, and other liabilities	$10,100
Unearned conference, membership, and subscription revenue	22,758
Total liabilities	**$32,858**
Net assets:	
Unrestricted	52.329
Temporarily restricted	5,721
Total net assets	**58,050**
Total liabilities and net assets	**$90,908**
Optional contributions fund – program expense	($000)
Education board accreditation	$95
USACM Committee	20
Total expenses	**$115**

Electronic Community

Communications of the ACM (http://cacm.acm.org/) Web site garnered the top award for Best New Web site by *Media Business*. The magazine's special report on "Ten Great Media Web Sites" recognized *Communications*' powerful search and browse functionality, deep integration with ACM's sizable archive of computing literature, and clean, fresh look.

ACM launched a new Multimedia Center last fall that offers members free access to select videos from various areas of interest in computing as well as from some of the organization's most popular activities and events. The Multimedia homepage (http://myacm.acm.org/dashboard.cfm?svc=mmc) features a collection of 10 videos at all times, with a new video replacing an existing one each week.

ACM is now providing its institutional library customers advanced electronic archiving services to preserve their valuable electronic resources. These services, provided by Portico and CLOCKSS, address the scholarly community's critical need for long-term solutions that assure reliable, secure, deliverable access to their digital collection of scholarly work. ACM is offering these services to protect the vast online collection of resources in its Digital Library used by over one million computing professionals worldwide.

ACM-W unveiled a new Web site this year that offers myriad ways to celebrate, inform, and support women in computing. The redesigned site (http://women.acm.org/) has many new features, including "Women of Distinction," highlighting women leaders; international activities of ACM-W Ambassadors and Regional Councils; and ways to get involved in attracting more young women to the computing profession.

It was a year that saw ACM—as well as most of its SIGs—establish a pres-

Statement of Activities: Year ended June 30, 2010 (in Thousands)

REVENUE	Unrestricted	Temporarily Restricted	Total
Membership dues	$9,201		$9,201
Publications	17,361		17,361
Conferences and other meetings	24,933		24,933
Interests and dividends	1,707		1,707
Net appreciation of investments	2,442		2,442
Contributions and grants	2,882	$963	3,845
Other revenue	348		348
Net assets released from restrictions	1,057	(1,057)	0
Total Revenue	**59,931**	**(94)**	**59,837**
EXPENSES			
Program:			
Membership processing and services	$945		$945
Publications	11,457		11,457
Conferences and other meetings	25,035		25,035
Program support and other	7,269		7,269
Total	**44,706**		**44,706**
Supporting services:			
General administration	9,009		9,009
Marketing	1,299		1,299
Total expenses	**55,014**		**55,014**
Increase (decrease) in net assets	4,917	(94)	4,823
Net assets at the beginning of the year	47,412	5,815	53,227
Net assets at the end of the year	**$52,329***	**$5,721**	**$58,050***

* Includes SIG Fund balance of $28,448K

ence on such popular social networks as Facebook, Twitter, and LinkedIn. SIGUCCS established an online community using Ning's social networking services and linked its portal to its new Web site (http://www.siguccs.org/) as well as initiated a series of Webinars to continue on a quarterly basis. SIGSIM's Modeling and Simulation Knowledge Repository (http://www.acm-sigsim-mskr.org) has proven an innovative program for supplying services to the SIGSIM technical community. And SIGMOBILE sponsored programs in the mobile computing research community such as a Community Resource for Archiving Wireless Data at Dartmouth (CRAWDAD). This wireless network, created to bridge the gap between research and real-world use of wireless networks, has rapidly become one of the most critical wireless network data resources for the global research community.

Conferences

SIGGRAPH 2009 welcomed 11,000 artists, research scientists, gaming experts, and filmmakers from 69 countries to New Orleans. Exhibits at SIGGRAPH experienced the largest percentage of international participation in more than 10 years, with a total of 140 industry organizations represented. In addition, over 965 speakers participated in the conference. SIGGRAPH Asia 2009 attracted over 6,500 visitors from more than 50 countries across Asia and globally to Yokohama, Japan where over 500 artists, academics, and industry experts shared their work.

ACM's SIG Governing Board agreed to sponsor select conferences that come to ACM without a technical tie to one its SIGs. In FY10, SGB approved sponsorship for two conferences: The ACM International Conference on Bioinformatics and Computational Biology and the First ACM International Health Informatics Symposium.

Attendance for ASSETS 09, sponsored by SIGACCESS, exceeded all projections, drawing a record number of participants to its technical program that addressed key issues such as cognitive accessibility, wayfinding, virtual environments, and accessibility obstacles for the hearing impaired.

SIGOP's flagship ACM Symposium on Operating Systems Principles enjoyed record-breaking attendance; the SIG also jointly sponsored (with SIGMOD) the first annual ACM Symposium on Cloud Computing.

KDD 2009 maintained SIGKDD's position as the leading conference on data mining and knowledge discovery, with a record number of submissions.

Recognition

The ACM Fellows Program, established in 1993 to honor outstanding ACM members for their achievements in computer science and information technology, inducted 47 new fellows in FY10, bringing the total number of ACM Fellows to 722.

ACM also recognized 84 Distinguished Members for their individual contributions to both the practical and theoretical aspects of computing and information technology. In addition, 150 Senior Members were recognized for demonstrated performances that set them apart from their peers.

There were 104 new ACM chapters chartered in last year. Of the 28 new professional chapters, 26 of them were internationally based; of the 76 new student chapters, 41 of them were based internationally.

BLOG@CACM

The *Communications* Web site, http://cacm.acm.org, features more than a dozen bloggers in the BLOG@CACM community. In each issue of *Communications*, we'll publish selected posts or excerpts.

twitter

Follow us on Twitter at http://twitter.com/blogCACM

DOI:10.1145/1866739.1866743 http://cacm.acm.org/blogs/blog-cacm

Smart Career Advice; Laptops as a Classroom Distraction

Jack Rosenberger shares Patty Azzarello's life lessons about advancing in the workplace. Judy Robertson discusses students' in-class usage of laptops.

Jack Rosenberger
"Are You Invisible?"
http://cacm.acm.org/blogs/blog-cacm/94307

Early in her career as a manager at Hewlett-Packard (HP), Patty Azzarello was put in charge of a software development team whose product life cycle took two years. It was "a ridiculously long period of time" to develop software, Azzarello says, and the length of the development cycle left both HP's sales team and its customers unhappy and frustrated. Azzarello revamped the team, reinvented its operating mode, and reduced the software cycle to nine months. The successful delivery of the new software life cycle's first product coincided with Azzarello's annual performance review, and she expected to receive a healthy raise as the economy was strong, HP was performing well, and Azzarello herself was awarding significant raises to her top employees.

Azzarello's own raise, however, was zero.

When Azzarello asked her boss why she wasn't receiving a raise, he replied, "Because nobody knows you."

Azzarello told this anecdote during her keynote speech at a DAC 2010 career workshop titled "More Than Core Competence...What it Takes for Your Career to Survive, and Thrive!" She learned from the experience, and proceeded to flourish at HP, becoming the youngest HP general manager at the age of 33, running a $1 billion software business at 35, and becoming a CEO at 39, she says. Today, Azzarello runs her own business management company, Azzarello Group, and gives career advice, which brought her to DAC.

Like Azzarello early in her career at HP, many employees believe that if they do their job and work hard, they will be recognized and justly rewarded. Not so, says Azzarello.

Azzarello's career advice for employees is to make sure the work that you do is aligned with the company's goals, bring your accomplishments to the attention of your superiors, and create a network of mentors who will guide you and ensure you win a spot on the company's list of employees who are the leaders of tomorrow. Her advice is tersely described as: Do better, look better, and connect better.

Do Better. "The most reliable way to advance your career is to add more value to your business," says Azzarello. You need to understand what is most important to your company in terms of your job—whether it's cutting costs, increasing Web traffic, or improving the delivery of software products—and focus on that. Everything else is less important.

However, too many employees, like the younger Azzarello, fall into the trap of being "workhorses." They do everything they're asked to do, and often more, but the end result is "you're being valued as a workhorse, not as a leader," Azzarello says.

To get ahead in the workplace, Azzarello says it's important to figure out how to deliver your work but also how to create free time at work during which you can manage and advance your career. After all, if you spend all of your time working, you won't have the time or energy to understand the company and its goals (which can change), promote yourself and your accomplishments, and build relationships with mentors and fellow employees.

The most successful employees learn how to be the master of their work and not let it control them. They understand which aspects of their job are most important to their company, and focus on them. "It's essential to be ruthless with your priorities," Azzarello says.

"Refuse to let your time get burned up with things that are less important."

A critical lesson Azzarello learned in her career at HP is that the "most successful executives don't do everything. They do a few things right and hit them out of the park."

Look Better. The second step of Azzarello's career plan involves making your work and accomplishments known to your immediate bosses. After all, if you deliver excellent results, but no one above you in the company is aware of them or doesn't connect the results with your job performance, it'll be difficult for you to advance in your company.

Azzarello recommends creating an audience list of the people in your company who should know about your achievements at work and a communication plan for how to inform these key players about your work and what you've accomplished. The audience list should include the influencers who have a say in your career—your bosses and your bosses' bosses—and any stakeholders who are dependent on your work. And your communication plan should describe how you will inform these influencers—usually via conversations, reports, and email—about your job and what you've accomplished.

For your achievements to be appreciated, Azzarello says it's vital that they are relevant to your company's goals. "Your priorities must be relevant to their priorities," says Azzarello. "Your work must be recognized as matching the business's goals."

Connect Better. The third step of Azzarello's career plan involves connecting with key players at your company, which involves building relationships with mentors and creating a broad network of support. "Successful people get a lot of help from others," Azzarello says. "You can't be successful alone."

Azzarello stresses the importance of mentors (note the plural) at your company and outside of it, and says employees "shouldn't attempt career advancement without mentors." Not only can mentors help you understand a company's culture and goals, but they, and other key players, can help you get a spot on the company's list of employees who are viewed as up and coming.

All of this is about visibility. The company president or other top executives must know or know about you, or you must have a relationship with mentors or others who are connected to the company president and key executives.

This step involves networking, and many people (and Azzarello admits she's one of these people) are uncomfortable with meeting new people for the purpose of networking. If you're one of these people, Azzarello's advice is to network with the people you already know.

If Azzarello's career advice sounds like a lot of work, you're right—it is. Which is why she urges employees to create a yearlong plan for implementing these three stages.

For many employees, Azzarello's advice is a real challenge. The alternative, however, is rather unsatisfying. After all, who wants a zero raise?

Judy Robertson "Laptops in the Classroom"

http://cacm.acm.org/blogs/blog-cacm/93398

Do you ever find yourself checking your email during a boring meeting? Do you drift off on a wave of RSS feeds when you should be listening to your colleagues? Do you pretend to be taking studious notes during seminars while actually reading Slashdot? In fact, shouldn't your full attention be somewhere else right now?

I find it increasingly tempting to do lots of things at once, or at least take microbreaks from activities to check mail or news. I do think it's rude to do so during meetings so I try to stop myself. My students don't tend to have such scruples. They use their laptops openly in class, and they're not all conscientiously following along with my slides, I suspect. In fact, in a recent study, "Assessing Laptop Use in Higher Education Classrooms: The Laptop Effectiveness Scale," published in the *Australasian Journal of Educational Technology*, 70% of students spent half their time sending email during class (instant messaging, playing games, and other nonacademic activities were also popular). They did also take notes and other learning tasks, but they weren't exactly dedicated to staying on task. If you're interested in surveying your own class to find out what they really do behind their screens, the study's authors provide a reliable, validated questionnaire tool about laptop usage in education.

Of course, there have always been distractions during class—as the margins of my Maths 101 notes demonstrate with their elaborate doodles. It's just that laptops make it so easy and seductive to drop your attention out of the lecture while still feeling that you are achieving something. ("I simply must update Facebook now. Otherwise people will not know I am in a boring lecture.")

Unsurprisingly, laptop usage in class has been associated with poorer learning outcomes, poorer self-perception of learning, and students reporting feeling distracted by their own screen as well as their neighbors' (see Carrie Fried's "In-class Laptop Use and its Effects on Student Learning"). Many educators get frustrated by this (see Dennis Adam's "Wireless Laptops in the Classroom [and the Sesame Street Syndrome]") and there is debate about whether laptops should be banned, or whether the lecturer should have a big red button to switch off wireless (or to electrocute all students) when he or she can't stand it anymore.

Bear in mind, though, the studies I mention here were conducted in lecture-style classes and the students were not given guidance on how to effectively use their laptops to help them learn rather than arrange their social lives. It is possible to design active classes around laptop use (if you can make sure that students who don't own a laptop can borrow one) thereby making the technology work in your favor. For example, my students learn to do literature searches in class, try out code snippets, or critique the design of Web pages. And, yes, some of them still get distracted from these activities and wander off to FarmVille. But at least I have given them the opportunity to integrate their technology with their learning in a meaningful way. They are adult learners after all. It's their decision how best to spend their brain cells in my class and my job is to give them a compelling reason to spend them on computer science rather than solitaire.

Jack Rosenberger is senior editor, news, of *Communications*. **Judy Robertson** is a lecturer at Heriot-Watt University.

DOI:10.1145/1866739.1866744 David Roman

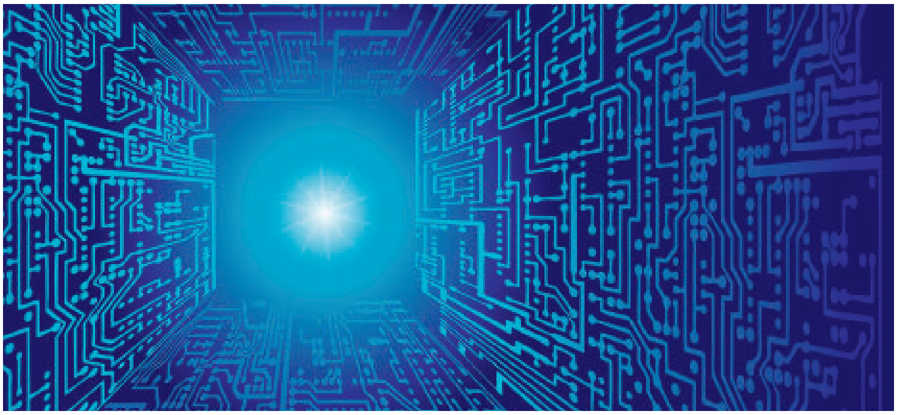
Scholarly Publishing Model Needs an Update

Science demands an overhaul of the well-established system of peer-review in scholarly communication. The current system is outmoded, inefficient, and slow. The only question is how!

The speed of scientific discovery is accelerating, especially in the field of computing, with an increasing number of ways to communicate results to global research communities, and to facilitate the exchange of ideas, critiques, and information through blogs, social networks, virtual meetings, and other electronic media in real time. These changes represent an enormous opportunity for scientific publishing.

Technology facilitated this acceleration, but technology alone will not provide the solution. Scientific discovery will not reduce or replace the need for good judgment, expertise, and quality should always take priority over speed. At times, these values are at odds with the speed of digital communication, and this is never more apparent than when spending a few spare moments reading general Twitter or Facebook posts in response to serious scholarly articles published online in established publications. The combination of social networking and scientific peer review is not a de facto home run.

Nevertheless, if implemented well, technology can help to serve as a springboard for positive changes to the scholarly communication process. But it's not clear how to measure the import or impact of these activities, or their ability to truly change the current system, which is still heavily dependent on a long established system of "publish or perish" in scholarly journals or conference proceedings. Many of the ways in which we communicate scientific discovery or conduct discourse are simply not counted in professional assessments, and this provides a negative incentive to changing the present system. The existing model of peer review is part of the problem, but the social system of rewarding only the long-established scholarly media (print/online journals and conference proceedings in the case of computer science) is also a major hurdle. The publication media that are accepted by the academic establishment happen to be those that take the most time to reach their intended readership. It is also worth noting that these media have stood the test of time. Science [CONTINUED ON P. 96]



ACM Member News

JAN CAMENISCH WINS SIGSAC'S OUTSTANDING INNOVATION AWARD

Jan Camenisch, a research staff member and project leader at IBM Research-Zurich, is the recipient of ACM's Special Interest Group on Security, Audit, and Control's (SIGSAC's) Outstanding Innovation Award. Camenisch was recognized for outstanding theoretical work on privacy-enhancing cryptographic protocols, which led to IBM's Identity Mixer, a system that authenticates a person's identity while preserving their privacy.

With colleagues, Camenisch has addressed the problem of preserving privacy in distributed systems, which often require a user to disclose more personal information than is necessary to gain access to online resources. For instance, Camenisch has developed cryptographic tools that allow a person to create a pseudonym for a subscription-based Web site. The cryptographically secure pseudonym proves the person has a subscription, but doesn't reveal any information about the individual's identity. It can be used in nearly all situations that require authentication, such as transacting business online with a smart card.

As technical leader of the European Union-funded Privacy and Identity Management for Europe project, which aims to give users more awareness of and control over their personal information, Camenisch is in the process of developing an entire identity management system.

He says many users of Facebook and other social media sites do not realize the extent of the footprints they are leaving on the Web, and that system designers don't put enough emphasis on identity protection.

His advice for prospective security experts? "Get fascinated by the cryptography and believe that you can solve seemingly paradoxical problems," Camenisch says. "And, of course, come work for IBM."

—*Neil Savage*

IMAGE COURTESY OF IBM RESEARCH - ZURICH

Science | DOI:10.1145/1866739.1866745 Gary Anthes

Nonlinear Systems Made Easy

Pablo Parrilo has discovered a new approach to convex optimization that creates order out of chaos in complex nonlinear systems.

IMAGINE YOU ARE hiking in a complex and rugged environment. You are surrounded by hills and valleys, mountains, shallow ditches, steep cliffs, and lakes. Nothing about the ground immediately around you, or your current direction, tells you much about where you will end up or what might lie in between.

Now imagine you are walking on the inside surface of a huge, smoothly shaped bowl. You can see the bottom of it, and even a few steps over the surface of the bowl tell you much about its shape and dimensions. There are no surprises.

Pablo Parrilo, a professor of electrical engineering and computer science at Massachusetts Institute of Technology (MIT), has found a way to remake the mathematical landscapes of complex, nonlinear systems into predictable smooth bowls. He has constructed a rare bridge between theoretical math and engineering that extends the frontiers of such diverse disciplines as chip design, robotics, biology, and economics.

Nonlinear dynamical systems are inherently difficult, especially when they involve many variables. Often they act in a linear fashion over some small region, then change radically in some other region. Water expands linearly as it warms, then explodes in volume at the boiling point. An airplane rises smoothly and ever more steeply—until it stalls. Understanding these systems often requires a great deal of prior knowledge, plus a painstaking combination of trial and error and modeling. Sometimes the models themselves are so complex their behavior can't be predicted or guaranteed, and running realistic models can be computationally intractable.

New algorithms devised by Pablo Parrilo, an MIT professor of electrical engineering and computer science, have made working with nonlinear systems both easier and more efficient.

PHOTOGRAPH BY PATRICK GILLOOLY

Parrilo developed algorithms that take the complex, nonlinear polynomials in models that describe these systems and—without actually solving them—rewrites them as much simpler mathematical expressions represented as sums of squares of other functions. Because squares can only be positive, his expressions are guaranteed to be greater than zero—the bottom of a "bowl"—and relatively straightforward to analyze via conventional mathematical tools such as optimization techniques.

Parrilo's transformed equations are "convex," like bowls. Convexity in mathematics essentially means that a function is free of undulations that form local minima and maxima. "It means that if you know something about two points, then you know what's going to happen in the middle," Parrilo says.

"The reason that the convexity property is so important is that it allows us to make global statements from local properties," Parrilo says. "You can sometimes give bounds on the quantity you are trying to find; essentially, you use the convexity of a function as a way of establishing whatever conclusion you want to make." In other words, one can deduce a great deal about a bowl from visiting just a few points on it.

Once Parrilo has derived the sums-of-squares equations, the equations are solved (a minimum is found) via an optimization technique called semidefinite programming, a relatively recent extension of linear programming that works on matrices representing convex functions. (The algorithms for both steps are contained in a MATLAB toolbox called Sostools, which is available at http://www.mit.edu/~parrilo/sostools/.)

While solving the original nonlinear equations is often NP-hard, Sostools can find useful bounds or even exact solutions from the convex sums-of-squares equations in polynomial time, Parrilo says. In fact, his techniques can improve efficiency so greatly—for researchers as well as their computers—that they enable fundamentally new ways of working and, in some cases, qualitatively better results.

Systems, Functions, Properties

Familiar systems—one describing energy, for example—are often defined by functions in which equilibrium exists at some minimum point, with the systems moving toward that point along smooth trajectories. "But with many systems, like a biological system, it is very different. We don't quite know what these functions are," Parrilo says. "So what [my] methods do, in a more or less automatic way, is find a function that has the properties of an energy function."

Because nonlinear systems are so difficult, system designers and researchers often take the easy but wrong way out, says Elizabeth Bradley, a professor of computer science at the University of Colorado at Boulder. "Linear systems dominate our education as engineers solely because they are easy," Bradley says. "But that leads to the lamppost problem. People look around the linear lamppost even though the answers aren't there." Parrilo's contribution is important, she says, because it makes dealing with nonlinear systems much easier.

John Harrison, a principal engineer at Intel, knows what it's like to wrestle with nonlinear systems. He develops formal proofs for the correctness of designs for floating-point arithmetic circuits. The idea is to prevent a recurrence of the floating-point division bug in the Pentium chip that cost Intel nearly $500 million in the mid-1990s. The tools he had used to do that, before discovering Parrilo's sums of squares and semidefinite programming, were complex, time consuming, and required huge amounts of computer time, he says. Now his formal verifications typically run in "tens of seconds" rather than "many minutes or even hours," Harrison says.

Harrison doesn't have to find exact solutions to his equations, but can work with proofs that a polynomial will remain within certain acceptable bounds over a specified range. That's exactly what Parrilo's method does, he says. "The key," he adds, "is the ability to certify the result formally, otherwise various less rigorous methods could be used."

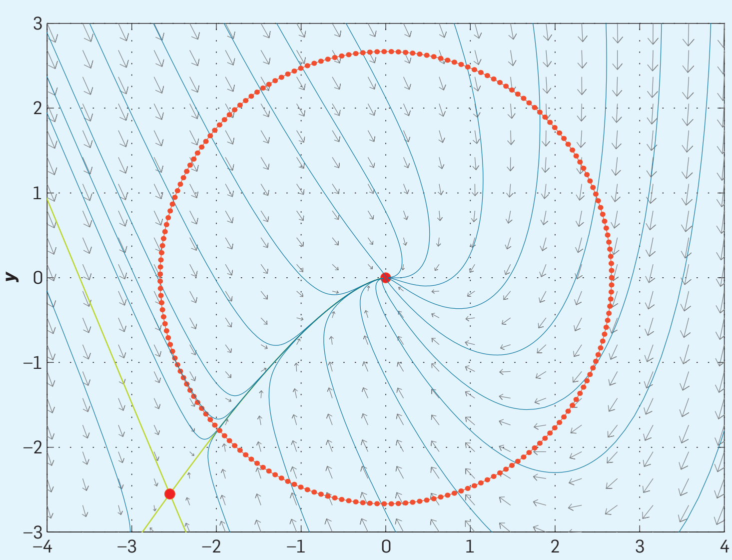
The broad scope of Parrilo's concepts may mean that formal methods, which can mathematically prove or verify the correctness of designs, but usually with some difficulty, will propagate more widely, Harrison predicts.

Specifying a Robot's Bounds

A robot walking slowly can be controlled by a relatively simple system that works linearly, says Russ Tedrake, associate professor of electrical engineering and computer science at MIT. But if the robot walks too fast or encounters some kind of disturbance, nonlinear factors kick in and the robot's behavior becomes much more difficult to predict and control. Tedrake is using Parrilo's sums-of-squares and semidefinite programming techniques to rigorously specify the bounds within which the robot won't fall. He has done the same to specify when a linear control system for a flying robot will become unable to keep the machine on its desired flight path.

Tedrake builds models of his robots' flight consisting of very complex differential equations. From those, there are well-established tools for defining workable flight paths, and given those trajectories, there are good linear systems for controlling the robot even when it devi-

Phase plot of a two-dimensional dynamical system, and estimate of the region of attraction of the stable equilibrium at the origin. This estimate was obtained by solving a sum of squares optimization problem.



ates from the desired trajectory. But if it veers too far off, nonlinear terms can take over and the robot may crash. But now, using Parrilo's tools, Tedrake can generate "proofs of stability" for these nonlinear terms that in essence define an envelope within which the robot will converge back to its nominal trajectory. "Proving stability in this way is important for the techniques to be accepted in many applications," he says.

"This opens up a rigorous way of thinking of nonlinear systems in ways not possible a few years ago," Tedrake says. "It used to be you had to be very innovative and creative to come up with a proof of stability for nonlinear systems." Now, he says, his results are more reliable and rigorous, and he can more easily devise and evaluate alternate flight paths and control systems.

Like Harrison, Tedrake hails an efficiency breakthrough in Parrilo's methods. Instead of laboriously verifying the stability of thousands of isolated trajectories, he can now work with just a few "regions of stability," he says.

Parrilo's work is noteworthy for the specific techniques and algorithms he has developed, but, at a higher level, it is impressive for its reach, says John Doyle, a professor of control and dynamical systems at the California Institute of Technology. Doyle points out that researchers have labored for decades to understand and control complex nonlinear systems, such as those that run computers, manage networks, and guide airplanes. Many useful tools—such as formal verification of software and hardware—and disciplines—such as robust control—have emerged from this work. But, Doyle says, there has not been a "unified approach" to the problem of anticipating and preventing unintended consequences in systems that are "dynamic, nonlinear, distributed and complex."

Now, Parrilo has made a giant step toward developing just such a unified approach, Doyle says. "What Pablo said was, 'Here is a systematic way to pursue these problems and, oh, by the way, a lot of these tricks that you guys have come up with over 30 years—in a whole bunch of fields that we didn't see as related—are all special cases of this strategy.' What he did is connect the dots."

Parrilo enables one to "automate the search for proofs," Doyle says. He explains it this way: "You have a set of bad behaviors that you don't want the system to have, and you have a model of the system. You want to prove that the two sets don't intersect. Pablo recognized that there was a sort of universal way to attack this set non-intersection problem."

As for the breadth of Parrilo's thinking, Doyle says, "The mathematicians think of Pablo as one of their own, and so do the engineers."

Pablo Parrilo has constructed a bridge between theoretical math and engineering that extends the frontiers of chip design, robotics, biology, and economics.

Further Reading

Boyd, S. and Vandenberghe, L.
***Convex Optimization*, Cambridge University Press, Cambridge, U.K, 2004.**

Dekker, S.
***Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization (Ph.D. dissertation)*, California Institute of Technology, Pasadena, CA, May 2000.**

Parrilo, P.A.
Semidefinite programming relaxations for semialgebraic problems, *Mathematical Programming Ser. B 96*, 2, 2003.

Parrilo, P.A.
Sum of squares optimization in the analysis and synthesis of control systems, 2006 American Control Conference, Minneapolis, MN, June 14–16 2006.

Parrilo, P.A. and Sturmfels, B.
Minimizing polynomial functions, Cornell University Library, March 26, 2001, http://arxiv.org/abs/math.OC/0103170.

Gary Anthes is a technology writer and editor based in Arlington, VA.

Education

Hispanics and STEM

Federal and state government agencies have taken steps to help more Hispanic students be trained in the science, technology, engineering, and mathematics (STEM) fields in the U.S., but Hispanic students remain severely underrepresented among STEM master's and doctoral degree recipients. Now, a report from the Center for Urban Education, *Tapping HSI-STEM Funds to Improve Latina and Latino Access to STEM Professions*, offers suggestions about improving Hispanics' STEM participation, and notes that financial issues often play an important role.

"It is clear that every computer scientist, scientist, and engineer has a role to play in diversifying the STEM fields and that the time to do it is now," says Alicia Dowd, associate professor of higher education at the University of Southern California and codirector of the Center for Urban Education.

The report alerts STEM administrators and faculty at Hispanic-Serving Institutions (HSIs) that substantial funds will be available over the next decade to support Hispanic STEM students, and provides recommendations for how to best use those funds.

HSIs can improve the number of Hispanic students earning STEM degrees by helping them balance their reliance on loans and earnings with grants and scholarships, the report notes. It recommends that when applying for HSI-STEM funds, HSIs should increase support for intensive junior- and senior-year STEM research experiences and propose programs that incorporate research opportunities into the core curriculum rather than into special programs that may not be accessible to working adults.

The report suggests that colleges, particularly those with large Hispanic populations, should inform students about their full range of financial-aid options. It also urges colleges to recognize that many Hispanic undergraduates are supporting themselves and are more likely to work than their peers.

—Bob Violino

Technology | DOI:10.1145/1866739.1866746 Alex Wright

The Touchy Subject of Haptics

After more than 20 years of research and development, are haptic interfaces finally getting ready to enter the computing mainstream?

EVER SINCE THE first silent-mode cell phones started buzzing in our pockets a few years ago, many of us have unwittingly developed a fumbling familiarity with haptics: technology that invokes our sense of touch. Video games now routinely employ force-feedback joysticks to jolt their players with a sense of impending on-screen doom, while more sophisticated haptic devices have helped doctors conduct surgeries from afar, allowed desk-bound soldiers to operate robots in hazardous environments, and equipped musicians with virtual violins.

Despite recent technological advances, haptic interfaces have made only modest inroads into the mass consumer market. Buzzing cell phones and shaking joysticks aside, developers have yet to create a breakthrough product—a device that would do for haptics what the iPhone has done for touch screens. The slow pace of market acceptance stems partly from typical new-technology growing pains: high production costs, the lack of standard application programming interfaces (APIs), and the absence of established user interface conventions. Those issues aside, however, a bigger question looms over this fledgling industry: What are haptics good for, exactly?

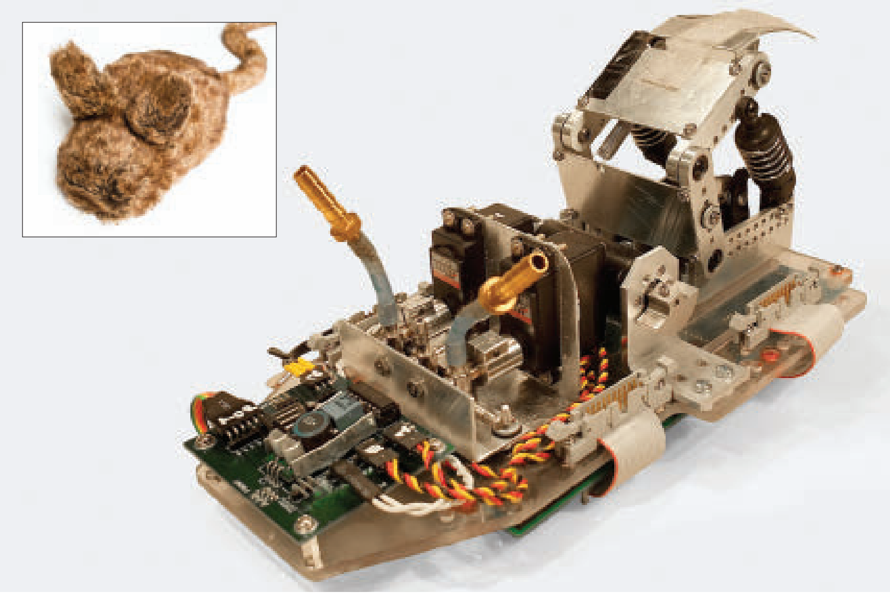
Computer scientists have been exploring haptics for more than two decades. Early research focused largely on the problem of sensory substitution, converting imagery or speech information into electric or vibratory stimulation patterns on the skin. As the technology matured, haptics found new applications in teleoperator systems and virtual environments, useful for robotics and flight simulator applications.

Today, some researchers think the big promise of haptics may involve moving beyond special-purpose applications to tackle one of the defining challenges of our age: information overload. For many of us, a growing reliance on screen-based computers has long since overtaxed our visual senses. But the human mind comes equipped to process information simultaneously from multiple inputs—including the sense of touch. "People are not biologically equipped to handle the assault of information that all comes through one channel," says Karon MacLean, a professor of computer science at the University of British Columbia.

Haptic interfaces offer the promise of creating an auxiliary information channel that could offload some of the cognitive load by transmitting data to the human brain through a range of vibrations or other touch-based feedback. "In the real world things happen on the periphery," says Lynette Jones, a senior research scientist at Massachusetts Institute of Technology. "It seems like haptics might be a good candidate for exploiting that capability because it's already a background sense."

As people consume more information on mobile devices, the case for haptics seems to grow stronger. "As screen size has become smaller, there is interest in offloading some information that would have been presented visually to other modalities," says Jones, who also sees opportunities for haptic interfaces embedded in vehicles as early warning systems and proximity indicators, as well as more advanced applications in surgery, space, undersea exploration, and military scenarios.

While those opportunities may be real, developers will first have to overcome a series of daunting technical obstacles. For starters, there is currently no standard API for the various force feedback devices on the market, although some recent efforts have resulted in commercial as well as open source solutions for developing software for multiple haptic hardware platforms. And as haptic devices grow



About the size of a cat, the Haptic Creature produces different sensations in response to human touch. Insert: The Haptic Creature with furry skin.

PHOTOGRAPHS BY STEVE YOHANAN

more complex, engineers will have to optimize for a much more diverse set of sensory receptors in the human body that respond to pressure, movement, and temperature changes.

As the range of possible touch-based interfaces expands, developers face a further hurdle in helping users make sense of all the possible permutations of haptic feedback. This lack of a standard "haptic language" may prove one of the most vexing barriers to widespread market acceptance. Whereas most people have by now formed reliable mental models of how certain software interfaces should work—keyboards and mice, touchpads, and touch screens, for example—the ordinary consumer still requires some kind of training to associate a haptic stimulation pattern with a particular meaning, such as the urgency of a phone call or the status of a download on a mobile device.

The prospect of convincing consumers to learn a new haptic language might seem daunting at first, but the good news is that most of us have already learned to rely on haptic feedback in our everyday lives, without ever giving it much thought. "We make judgments based on the firmness of a handshake," says Ed Colgate, a professor of mechanical engineering at Northwestern University. "We enjoy petting a dog and holding a spouse's hand. We don't enjoy getting sticky stuff on our fingers." Colgate believes that advanced haptics could eventually give rise to a set of widely recognized device behaviors that go well beyond the familiar buzz of cell phones. For now, however, the prospect of a universal haptic language seems a distant goal at best.

"Until we have a reasonably mature approach to providing haptic feedback, it's hard to imagine something as sophisticated as a haptic language arising," says Colgate, who believes that success in the marketplace will ultimately hinge on better systems integration, along the lines of what Apple has accomplished with the iPhone. "Today, haptics is thought of as an add-on to the user interface," says Colgate. "It may enhance usability a little bit, but its value pales in comparison to things you can do with graphics and sound. In many cases, the haptics is so poorly implemented that people turn it off pretty quickly. And that's not to criticize the developers of haptics—it's just a tough problem."

As haptic devices grow more complex, engineers will have to optimize for a much more diverse set of sensory receptors in the human body that respond to pressure, movement, and temperature changes.

Many efforts to date have used haptics as a complementary layer to existing screen-based interfaces. MacLean argues that haptics should do more than just embellish an interaction already taking place on the screen. "A lot of times you're using haptics to slap it on top of a graphical interaction," she says. "But there can also be an emotional improvement, a comfort and delight in using the interface."

Led by Ph.D. candidate Steve Yohanan, MacLean's team has built the Haptic Creature, a device about the size of a cat that simulates emotional responses. Covered with touch sensors, the Haptic Creature creates different sensations—hot, cold, or stiffening its "ears" in response to human touch. The team is exploring possible applications such as fostering companionship in older and younger people, or treating children with anxiety disorders.

MacLean's team has also developed an experimental device capable of buzzing in 84 different ways. After giving users a couple of months to get familiar with the feedback by way of an immersive game, they found that the process of learning to recognize haptic feedback bore a great deal of similarity to the process of learning a language.

HPC

Students Build Green500 Supercomputer

A team of students at the University of Illinois at Urbana-Champaign (UIUC) have built an energy-efficient supercomputer that appeared on both the Green500 and Top500 lists. Named in honor of one of the UIUC campus's main thoroughfares, the Green Street supercomputer placed third in the Green500 list of the world's most energy-efficient supercomputers, with a performance of 938 megaflops per watt. It also placed 403rd in the Top500 list, a ranking of the world's fastest supercomputers, with a performance of 33.6 teraflops.

The Green Street supercomputer grew out of an independent study course led by Bill Gropp, the Bill and Cynthia Saylor Professor of Computer Science, and Wen-mei Hwu, who holds the AMD Jerry Sanders Chair of Electrical and Computer Engineering. Approximately 15 UIUC undergraduate and graduate students helped build the supercomputer, which boosts a cluster of 128 graphics processing units donated by NVIDIA, and uses unorthodox supercomputer building materials, such as wood and Plexiglas.

The UIUC team hopes to increase the supercomputer's energy efficiency by 10%–20% with better management of its message passing interface and several other key elements. "You really need to make sure that the various parts of your communications path, in terms of different software layers and hardware drivers and components, are all in tune," says Hwu. "It's almost like when you drive a car, you need to make sure that all these things are in tune to get the maximum efficiency."

The Green Street supercomputer is being used as a teaching and research tool.

—Graeme Stemp-Morlock

"The surprising thing is that people are able to quickly learn an awful lot and learn it without conscious attention," says MacLean. "There's a lot of potential for people to learn encoded signals that mean something not in a representational way but in an abstract way without conscious attention."

To date, most low-cost haptic interfaces have relied exclusively on varying modes of vibration, taking advantage of the human skin's sensitivity to movement. But vibration constitutes the simplest, most brute-force execution of haptic technology. "Unfortunately," says Colgate, "vibration isn't all that pleasing a sensation."

Some of the most interesting research taking place today involves expanding the haptic repertoire beyond the familiar buzz of the vibrating cell phone. At MIT, Jones' team has conducted extensive research into human body awareness and tactile sensory systems, examining the contribution of receptors in the skin and muscles to human perceptual performance. In one study, Jones demonstrated that users were unable to distinguish between two thermal inputs presented on a single finger pad; instead, they perceived it as a single stimulus, demonstrating the tendency of thermal senses to create "spatial summation" rather than fine-tuned feedback.

Colgate's research has focused on a fingertip-based interface that provides local contact information using new actuation technologies including shear skin stretch, ultrasonic, and thermal actuators. By varying the friction in correspondence with fingertip motion across a surface, the interface can simulate the feeling of texture or a bump on the surface. Compared with force-feedback technology, vibrotactile stimulators, known as tactors, are much smaller in size and more portable, although high-performance tactors with wide bandwidths, small form factors, and independently controllable vibrational frequency and amplitude are still hard to come by at a reasonable cost.

The Northwestern researchers have figured out how to make transparent force sensors that can capture tactile feedback on a screen, so that they can be combined with a graphical display. "My ideal touch interface is one that can apply arbitrary forces to the finger," says Colgate, whose team has been approaching the problem by combining friction control with small lateral motions of the screen itself.

By controlling the force on the finger, the system can make parts of the screen feel "magnetic" so that a user's finger is pulled toward them—up, down, left, right—or letting a user feel the outline of a button on the screen where none exists. Colgate's team is also exploring how to develop devices using multiple fingers, each on a different variable friction interface.

Looking ahead, Colgate believes the evolution of haptic interfaces may follow the trajectory of touch screens: a technology long in development that finally found widespread and relatively sudden acceptance in the marketplace. "The technology has to be sufficiently mature and robust, there has to be an active marketplace that creates competition and drives down costs, and it has to meet a real need."

As production costs fall and new standards emerge—as they almost certainly will—the marketplace for touch-based devices may yet come into its own. Until that happens, most of the interesting work will likely remain confined to the labs. And the future of the haptics industry seems likely to remain, well, a touchy subject. ■

Further Reading

Chubb, E.C., Colgate, J.E., and Peshkin, M.A.
ShiverPaD: a glass haptic surface that produces shear force on a bare finger, *IEEE Transactions on Haptics 3*, 3, July–Sept., 2010.

Ferris, T.K. and Sarter, N.
When content matters: the role of processing code in tactile display design, *IEEE Transactions on Haptics 3*, 3, July–Sept., 2010.

Jones, L.A. and Ho, H.-N.
Warm or cool, large or small? The challenge of thermal displays, *IEEE Transactions on Haptics 1*, 1, Jan.–June, 2008.

MacLean, K.E.
Putting haptics into the ambience, *IEEE Transactions on Haptics 2*, 3, July–Sept., 2009.

Ryu, J., Chun, J., Park, G., Choi, S., and Han, S.H.
Vibrotactile feedback for information delivery in the vehicle, *IEEE Transactions on Haptics 3*, 2, April–June, 2010.

Alex Wright is a writer and information architect who lives and works in Brooklyn, NY. Hong Z. Tan, Purdue University, contributed to the development of this article.

Obituary

Watts Humphrey, Software Engineer: 1927–2010

Watts Humphrey, who distinguished himself as the "father of software quality engineering," died on October 28 at age 83 at his home in Sarasota, FL. Humphrey combined business practices with software development, and brought discipline and innovation to the process of designing, developing, testing, and releasing software.

"Watts had a profound impact on the field," says Anita Carleton, director of the Software Engineering Process Management Program at the Carnegie Mellon Software Engineering Institute (SEI). "He was a visionary, a wonderful leader, and a wonderful man."

After receiving B.S. and M.S. degrees in physics from the University of Chicago and the Illinois Institute of Technology, respectively, and an MBA from the University of Chicago, Humphrey went to work at IBM. There, he headed a team that introduced software licenses in the 1960s. Humphrey focused on how disciplined and experienced professionals, working as teams, could produce high quality, reliable software within committed cost and schedule constraints.

In 1986, after a 27-year career as a manager and executive at IBM, Humphrey joined SEI and founded the school's Software Process Program. He led the development of the Software Capability Maturity Model and eventually the Capability Maturity Model Integration (CMMI), a framework of software engineering best practices now used by thousands of organizations globally.

Humphrey was also the author of 11 books, including *Managing a Software Process*. An ACM and SEI Fellow, he was awarded the National Medal of Technology in 2005.

—*Samuel Greengard*

Society | DOI:10.1145/1866739.1866747 Marina Krakovsky

India's Elephantine Effort

An ambitious biometric ID project in the world's second most populous nation aims to relieve poverty, but faces many hurdles.

A 95-year-old Indian man has his fingerprints scanned as part of the Unique ID project.

DESPITE INDIA'S ECONOMIC boom, more than a third of the country remains impoverished, with 456 million people subsisting on less than $1.25 per day, according to the most recent World Bank figures. Government subsidies on everything from food to fuel have tried to spread the nation's wealth, but rampant corruption has made the redistribution pipeline woefully inefficient.

The "leakage" happens in part because the benefits aren't directed at specific individuals, says Salil Prabhakar, a Silicon Valley-based computer scientist who is working as a volunteer for the World Bank as part of the Unique ID (UID) project, a massive biometrics initiative aimed at overhauling the current system. "If I as the government issue $1 of a benefit," Prabhakar explains, "I don't know where it's going. I just know that there's a poor person in some remote village, and I hope it reaches them."

The current system relies on a chain of middlemen—many of them corrupt bureaucrats at various levels of government—who collectively siphon off 10% or more of what's due to the poor and resell the goods and services on the black market. For example, according to Transparency International, officials extracted $212 million in bribes alone from Indian households below the poverty line in 2007.

The UID project—helmed by the much-admired former Infosys Technologies CEO Nandan Nilekani and operated by Unique Identification Authority of India, a government agency—promises to be the first step in the solution. Recently renamed Aadhaar (meaning "foundation" in Hindi), the UID project plans to assign a unique 16-digit number to each citizen above the age of 18 who wants national identification, and to link that number with the owner's biometric data—all 10 fingerprints, an iris scan, and a headshot (plus four hidden "virtual" digits). Aadhaar's national enrollment was launched in September, with the goal of issuing 100 million ID numbers by March and 600 million within four years. Like the Social Security number in the U.S., the number won't guarantee government aid, but your biometrics will prove the UID is yours, letting you claim whatever benefits to which you're entitled. In theory, the result should be the end of counterfeit ration cards and other fraud, as well as making it easier for hundreds of millions of Indian adults to gain easier access to banking services for the first time. And because the system will work nationwide, Aadhaar should make it possible for the poor to move without losing benefits. The lower-income Indians love the idea, says Prabhakar, who witnessed what he describes as "almost a stampede" during a recent proof-of-concept enrollment.

The full implementation, though, is fraught with problems, most of which stem from the project's sheer size, given India's population of 1.2 billion. "Biometric systems have never operated on such a massive scale," says Arun Ross, an associate professor of computer science at West Virginia University.

One of the biggest challenges is deduplication. When a new user tries to enroll, the system must check for duplicates by comparing the new user's data against all the other records in the UID database. Hundreds of millions of records make this a computationally demanding process, made all the more so by the size of each record, which includes up to 12 higher-resolution images.

The demands continue each time there's an authentication request. "The matching is extremely computationally intensive," says Prabhakar. At peak times, the system must process tens of millions of requests per hour while responding in real time, requiring massive data centers the likes of Google's.

Achieving acceptable levels of accuracy at this scale is another major difficulty. Unlike passwords, biometrics never produce an exact match, so matching always entails the chance of false accepts and false rejects, but as the number of enrollments rises, so do the error rates, since it becomes more likely that two different individuals will share similar biometrics. Using a combination of biometrics—instead of a single thumbprint, for example—greatly improves accuracy and deters impostors. (In the words of Marios Savvides, assistant research professor in the department of electrical and computer engineering at Carnegie Mellon University, "It's hard to spoof fingerprints, face, and iris all at the same time.") But using multiple biometrics requires extra equipment, demands information fusion, and adds to the data processing load.

Other steps to improve accuracy also bring their own challenges. "The key issue," says Nalini Ratha, a researcher at the IBM Watson Research Center, "is have I captured enough variation so I don't reject you, and at the same time I don't match against everybody else?" Capturing the optimal amount of variation requires consistent conditions across devices in different settings—no easy feat in a country whose environment varies from deserts to tropics and from urban slums to far-flung rural areas. "It's almost like having many different countries in a single country, biometrically speaking," says Ross.

The challenge isn't just to reduce errors—under some conditions, a biometric reader may not work at all. "If it's too hot, people sweat and you end up with sweaty fingers," says Prabhakar, "and if it's too dry, the finger is too dry to make good contact with the optical surface of the scanner." Normalizing across varied lighting conditions is essential, since all of the biometric data is optical.

India's diverse population presents a whole other set of hurdles. Many of the poor work with their hands, but manual labor leads to fingertips so callused or dirty they can't produce usable fingerprints. And some of the most unfortunate residents are missing hands or eyes altogether.

"The key issue," says Nalini Ratha, "is have I captured enough variation so I don't reject you, and at the same time I don't match against everybody else?"

Security Challenges

As if these problems weren't enough, the UID system poses formidable security challenges beyond the threat of spoofing. "People get carried away by one type of attack—a fake finger, a fake mask, or something," says IBM's Ratha, "but there are probably 10 other attacks to a biometric system that can compromise the system."

For starters, when data is stored in a centralized database, it becomes an attractive target for hackers. Another vulnerability is the project's reliance on a network of public and private "registrars"—such as banks, telecoms, and government agencies—to collect biometric data and issue UIDs. Though registrars might ease enrollment, they're not necessarily worthy of the government's trust. Banks, for example, have been helping wealthy depositors evade taxes by opening fictitious accounts, so entrusting the banks with biometric devices doesn't make sense, says Sunil Abraham, executive director of the Centre for Internet and Society in Bangalore. "If I'm a bank manager, I can hack into the biometric device and introduce a variation in the fingerprint because the device is in my bank and the biometric is, once it's in the computer, just an image sent up the pipe," he says. Though careful monitoring could catch such hacks, Abraham says that's not realistic once you've got as many records as Aadhaar will have.

Registrars may also make UIDs, which are officially voluntary, a de facto requirement for services, especially in the current absence of a law governing how the data can be used. Such "function creep" troubles privacy advocates like Malavika Jayaram, a partner in the Bangalore-based law firm Jayaram & Jayaram, who says, "If every utility and every service I want is denied to me without a UID card, how is it voluntary?" The loss of civil liberties is too high a price to pay for a system that she believes leaves gaping opportunities for continued corruption. "The guy handing out the bags of rice could ask for a bribe even to operate the machine that scans the fingerprints, or he could say that the machine isn't working," says Jayaram. "And there's every chance the machine isn't working. Or he could say, 'I don't know who you are and I don't care; just pay me 500 rupees and I'll give you a bag of rice.' All the ways that humans can subvert the system are not helped by this scheme."

Abraham suggests a more effective way to root out fraud through biometrics would be to target the much smaller number of residents who own most of the country's wealth, much of it ill-gotten. "The leakage is not happening at the bottom of the pyramid," he says. "It's bureaucrats and vendors and politicians throughout the chain that are corrupt."

Despite all the technical and social challenges, Nandan Nilekani's UID project is on course to provide 100 million Indian residents with a Unique ID by March. Will Nilekani's UID scheme work? Only time will tell. "But if there's anybody in India who's capable of pulling it off, it's him," says Abraham. Meanwhile, the hopes of millions of India's poor are invariably tied to the project's success.

Further Reading

Bolle, R.M., Connell, J.H., Pankanti, S., Ratha, N.K., Senior, A.W.
Guide to Biometrics. **Springer, New York, NY, 2004.**

Jain, A.K., Flynn, P. and Ross, A. (Eds.)
Handbook of Biometrics. **Springer, New York, NY, 2007.**

Pato, J.N and Millett, L.I. (Eds.)
Biometric Recognition: Challenges and Opportunities. **The National Academies Press, Washington, D.C., 2010.**

Ramakumar, R.
High-cost, high-risk, *FRONTLINE 26*, 16, Aug. 1–14, 2009.

Ross, A. and Jain, A.K.
Information fusion in biometrics, *Pattern Recognition Letters 24*, 13, Sept. 2003.

Marina Krakovsky is a San Francisco area-based journalist and co-author of *Secrets of the Moneylab: How Behavioral Economics Can Improve Your Business.*

Milestones | DOI:10.1145/1866739.1866767

Jack Rosenberger

EMET Prize and Other Awards

Edward Felten, David Harel, Sarit Kraus, and others are honored for their contributions to computer science, technology, and electronic freedom and innovation.

THE A.M.N. FOUNDATION, Franklin Institute, Electronic Frontier Foundation, and other organizations recently recognized leading computer scientists and technologists.

EMET Prize

David Harel and **Sarit Kraus** were honored with EMET Prizes by the A.M.N. Foundation for the Advancement of Science, Art, and Culture in Israel for excellence in the computer sciences. Harel, a professor of computer science at the Weizmann Institute of Science, was recognized for his studies on a wide variety of topics within the discipline, among them logic and computability, software and systems engineering, graphical structures and visual languages, as well as modeling and analysis of biological systems.

Kraus, a professor of computer science at Bar-Ilan University, was recognized for her expertise in the field of artificial intelligence, along with her significant contributions to the field of autonomous agents, and studies in the field of multiagent systems.

FTC Chief Technologist

Edward Felten, a professor of computer science and public affairs at Princeton University, was named the U.S. Federal Trade Commission's first Chief Technologist. Felten, a vice-chair of ACM's U.S. Public Policy Council, will advise the federal agency on evolving technology-related issues of consumer protection, such as online privacy and cybersecurity, and antitrust matters, including tech industry mergers and anticompetitive behavior.

EFF Pioneer Awards

The Electronic Frontier Foundation presented its 2010 Pioneer awards to

Edward Felten, the U.S. Trade Commission's first Chief Technologist.

PHOTOGRAPH: PRINCETON UNIVERSITY, OFFICE OF COMMUNICATIONS, DENISE APPLEWHITE

four individuals who are extending freedom and innovation in the digital world. The honorees are:

Steven Aftergood, who directs the Federation of American Scientists Project on Government Secrecy, which works to reduce the scope of official secrecy and to promote public access to government information;

James Boyle, William Neal Reynolds Professor of Law and cofounder of the Center for the Study of the Public Domain at Duke Law School, who was recognized for his scholarship on the "second enclosure movement"—the worldwide expansion of intellectual property rights—and its threat to the public domain of cultural and scientific materials that the Internet might otherwise make available;

Pamela Jones, a blogger, and her Web site Groklaw, were honored for the creation of a new style of participatory journalism and distributed discovery, which has enabled programmers and engineers to educate lawyers on technology relevant to legal cases of significance to the Free and Open Source community, and which, in turn, has taught technologists about the workings of the legal system; and

Hari Krishna Prasad Vemuru, a security researcher in India, who revealed security flaws in India's paperless electronic voting machines, and endured jail time and political harassment to protect an anonymous source who enabled him to conduct the first independent security review of India's e-voting system.

Franklin Institute Laureate

John R. Anderson, R.K. Mellon University Professor of Psychology and Computer Science at Carnegie Mellon University, was named a 2011 Laureate by the Franklin Institute and awarded the Benjamin Franklin Medal in Computer and Cognitive Science "for the development of the first large-scale computational theory of the process by which humans perceive, learn, and reason, and its application to computer tutoring systems."

Prince Philip Designers Prize

Bill Moggridge, director of the Smithsonian Cooper-Hewitt National Design Museum, was awarded the 2010 Prince Philip Designers Prize, an annual award by the Duke of Edinburgh that recognizes a lifetime contribution to design, for his contributions to the GRiD Compass laptop. Released in 1982, the GRiD Compass is widely credited as being the forerunner of today's modern laptop.

Jack Rosenberger is senior editor, news, of *Communications.*

viewpoints

DOI:10.1145/1866739.1866748

Phillip G. Armour

The Business of Software
Don't Bring Me a Good Idea

How to sell process changes.

"You want to know how to get my attention?" Jason Kalich asked the audience rhetorically. "First off, don't bring me a good idea—I've already got plenty of good ideas." Kalich, the general manager of Microsoft's Relationship Experience Division, was participating in the keynote panel at the Quest Conference in Chicago.[a] The three industry experts on the panel were addressing the question asked by the moderator Rebecca Staton-Reinstein: *"How can I get my manager's buy in (to software quality and process change initiatives)?"* The audience consisted of several hundred software professionals, most of them employed in the areas of software quality, testing, and process management. Kalich had clearly given the topic a lot of thought and he warmed to the theme: "Don't even bring me cost savings. Cost savings are nice, but they're not what I'm really interested in." He paused for emphasis. "Bring me revenue growth and you've got my ear. Bring me new value, new products, new customers, new markets: then you've got my attention, then you've got my support. Don't bring me a good idea. Not interested."

a http://www.qaiquest.org/chicago/index.html

Sponsorship

Obtaining sponsorship for software development process changes is essential. The first of Watts Humphrey's Six Rules of Process Change is "start at the top"—get executive sponsorship for whatever change you are trying to make.[1] Without solid and continuing executive commitment to support changes they usually wither on the vine. But just how does a software quality professional get this support? Kalich and the other panelists were adamant that a good idea, even when supported by possible cost savings, just doesn't cut it in the current economic climate.

What the panel was saying is that good ideas are just that. And cost reduction, while valuable, tends to be quite incompressible—once the first 10%–20% of cost savings are achieved, further savings usually become increasingly difficult to get. Reducing costs is like compressing a spring—it may require more and more energy for less and less movement.

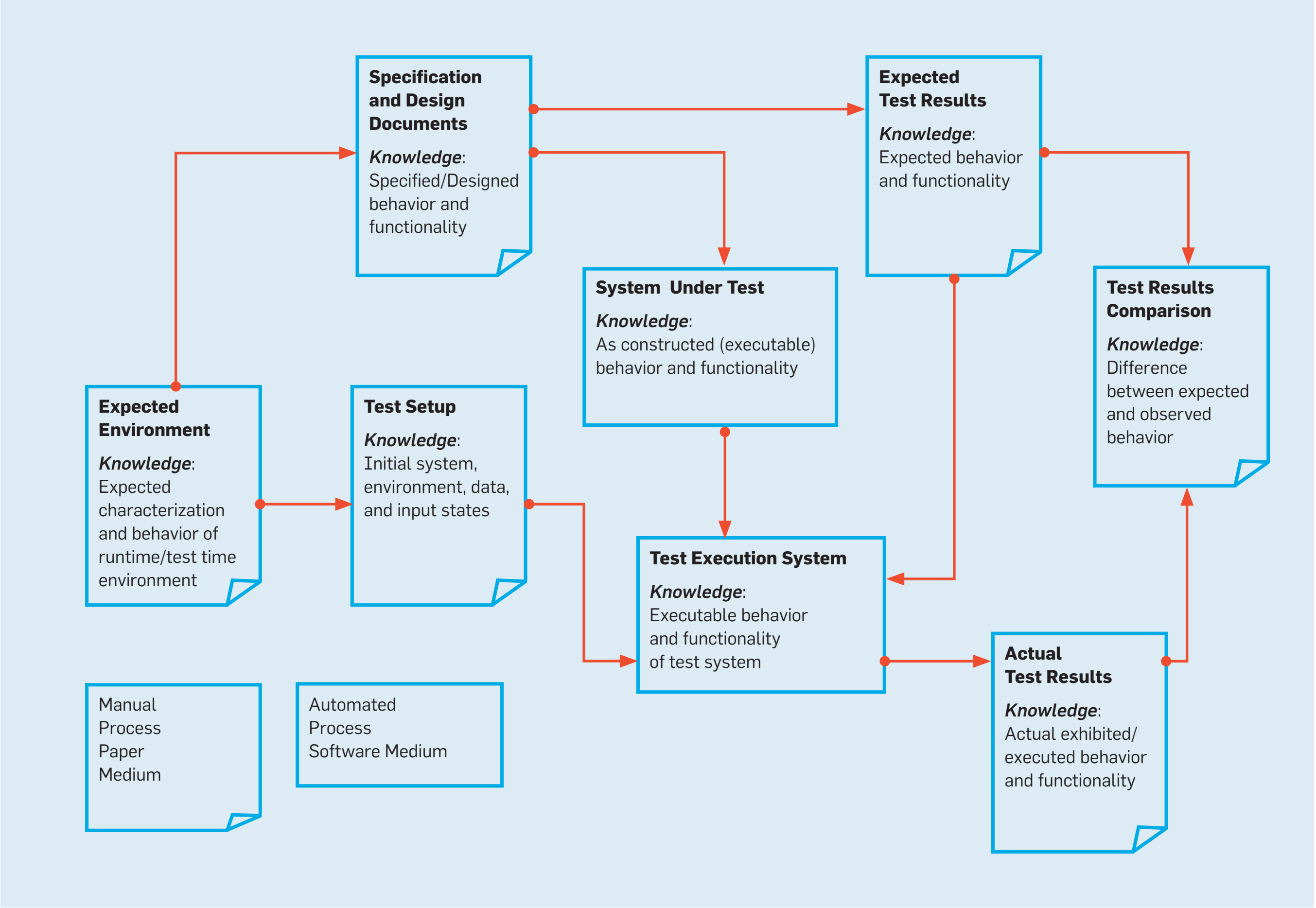
I looked around the audience and, while there were nods of understanding, there were also many blank stares as people tried to figure out: How can I turn my process initiative into a profit center? Making money is not a typical goal of process change as it is usually practiced which, according to the panel, might be why it doesn't always get the support it might.

But how to actually do it? That afternoon, I attended a presentation that showed how it can be done, and what critical success factors are needed to make it work.

SmartSignal

"Predictive Analytics is a really complex data set," said George Cerny, "our

Knowledge-containing artifacts involved in testing.



systems predict the possible failure of commercial aircraft, power stations, and oil rigs sometimes weeks before a failure might actually occur." Cerny is the quality assurance manager at SmartSignal,[b] an Illinois-based data analytics company.

To manage predictive analytics, large and complex systems must be instrumented and enormous amounts of complicated data must be collected from many different sources: pumps, power meters, pressure switches, maintenance databases, and other devices. Sometimes data is collected in real time, sometimes it is batched. Simple data is monitored for threshold conditions and complex interactive data is analyzed for combinational conditions. The analysis system must recognize patterns that indicate the future possibility of component, subsystem, or systemic failure and what the probability of that failure might be. And then it needs to report what it finds. Sometimes these reports are large and detailed; sometimes they are urgent and immediate.

"But before all this happens, the analytic system must be set up." Cerny said. "This setup was manual and data-entry intensive. A single power station might have hundreds of items of equipment that need to be monitored. Each item might have hundreds of measurements that must be taken over short, medium, and long timeframes. Each measurement might be associated with many similar or different measurements on the same device or on other equipment." The screen flashed with list after list of data items. "So how could we test this? How could we make sure the system works before we put it in?"

Testing a System

Testing is the interaction of several knowledge-containing artifacts, as shown in the accompanying figure. Some of these artifacts must be in executable software form, but many others are often in a paper format and are processed manually.

Cerny described this: "We realized early we had to test using virtual machines, but how could we test these? And how could we ensure scalability with both the numbers and the complexities of environments and inputs?" To the testing group at SmartSignal test automation was clearly a good idea. But how to get sponsorship for this good idea?

Jim Gagnard, CEO of SmartSignal, put it this way: "We are a software company whose products measure quality and everything is at risk if we aren't as good as we can be in everything we do. Leaders can help define and reinforce the culture that gets these results, but if it's not complemented with the right people who truly own the issues, it does not work."

Dave Bell, vice president of Application Engineering and Stacey Kacek, vice president of Product Development at SmartSignal, concurred. "We always have to be looking to replace

b http://www.smartsignal.com

what people do manually with the automated version of the same," said Kacek, "…once it works." Bell added: "While the management team understood the advantages of test automation and we all have an engineering background, we had to keep asking: *how to get buy-in*?" "Our driver was to find creative ways for our customers to make decisions, what we call '*speed to value*'" Kacek asserted.

Some Steps

Cerny described a few of the steps they took at SmartSignal to build their automated test system:

- Build to virtual machines and virtualization to isolate device dependence;
- Start simply using comma delimited scripts and hierarchical tree data views;
- Build up a name directory of functions;
- Separate global (run in any environment) from local variables;
- Keep object recognition out of scripts, use both static and dynamic binding; and
- Initially automate within the development team to prove the concept before moving to production.

Test Knowledge is System Knowledge

These steps are typical engineering design actions anyone might take in automating testing or, indeed, in automating any process or any system. But in this case there was a difference.

"Asset configuration is a big issue in the power industry." Cerny said in his presentation at Quest. "Imagine setting up a power station: what equipment should go where? Which pumps are used and connected to which other equipment? Where are sensors to be placed? What is the 'best' configuration of equipment that will most likely reduce the overall failure rate of the plant?"

The analytics test system is designed to prove the analytical system itself works. To do this, the test system must be set up (automatically, of course) to the appropriate target system configuration. The normal test function is meant to show that the analytical system will work as built for that particular target system. But what if we turn this around? What if we use our testing capability to find out *what configuration would show the lowest likely failure rate*? Doing this allows field engineers and power plant designers to model different configurations of systems for least likelihood of failure *before* they actually build and install them.

The knowledge of how to set up the test system is also the knowledge of how to set up the target production system.

The knowledge in the test system is the same as the knowledge in the target system. The knowledge of how to set up the test system is also the knowledge of how to set up the target production system. Automating this knowledge allows simulation of a system before it is built.

A *Really* Good Idea

This is what Jason Kalich and the panel at Quest were looking for. Automating the test system at SmartSignal ended up being not simply about speeding things up a bit, making the testers' lives easier, or saving a few dollars. It was not just about cranking through a few more tests in limited time or reducing test setup, analysis, and reporting time. It became something different—it became a configuration simulator and that's a new product.

If we automate knowledge in the right way, even internal software process knowledge, it can be used in many different ways and it can even be used to create new functionality and new products that our customers will pay for.

Now *that's* a good idea.

Reference

1. Humphrey, W. *Managing the Software Process.* Prentice Hall, New York, 1989, 54.

Phillip G. Armour (armour@corvusintl.com) is a senior consultant at Corvus International Inc., Deer Park, IL.

Calendar of Events

January 17–20
The Thirteenth Australiasian Computing Education Conference,
Perth QLD Australia,
Contact: Michael de Raadt,
Email: deraadt@usq.edu.au

January 22–26
Fifth International Conference on Tangible, Embedded, and Embodied Interaction,
Funchal, Portugal,
Contact: Mark D. Gross,
Email: mdgross@cmu.edu

January 23–29
The 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles
of Programming, Languages
Austin, TX,
Sponsored: SIGPLAN,
Contact: Thomas J. Ball,
Email: tball@microsoft.com

January 25–28
16th Asia and South Pacific Design Automation Conference,
Yokohama, Japan,
Sponsored: SIGDA,
Contact: Kunihiro Asada,
Email: asada@silicon.u-tokyo.ac.jp

January 28–30
Global Game Jam,
Multiple Locations (TBA),
Contact: Bellamy Gordon,
Email: gordon@igda.org

February 2–5
Fourth ACM International Conference on Web Search and Data Mining,
Kowloon, Hong Kong,
Sponsored: SIGKDD, SIGMOD, SIGWEB, SIGIR,
Contact: Irwin K. King,
Email: king@cse.cukhk.edu.hk

February 9–10
International Symposium on Engineering Secure Software and Systems,
Madrid, Spain,
Contact: Ulfar Erlingsson,
Email: ulfar@yahoo.com

February 9–12
Fourth ACM International Conference on Web Search and Data Mining,
Kowloon, Hong Kong,
Sponsored: SIGWEB, SIGIR, SIGKDD, and SIGMOD,
Contact: Irwin K. King,
Email: king@cse.cukhk.edu.hk

DOI:10.1145/1866739.1866749

Stefan Bechtold

Law and Technology

Google AdWords and European Trademark Law

Is Google violating trademark law by operating its AdWords system?

WHEN THE DOT-COM boom began in the late 1990s, many analysts and observers proclaimed the death of intermediation. Supply chains seemed to become shorter and shorter as new B2C companies emerged in Silicon Valley. These companies could deal with their customers directly over the Internet, rendering distributors, wholesalers, brokers, and agents superfluous.

While some traditional middlemen have indeed become less important as Internet commerce has developed, we have not seen a general death of intermediation. Rather, many new intermediaries have arisen on the digital landscape over the last 15 years. Just think of Amazon, eBay, or Google. If all these companies have been successful, it is not because they have removed all barriers between producers and consumers. They have been successful because they offer innovative services located between producers and consumers along the digital supply chain.

The law often has a difficult time coping with new intermediaries. Should an Internet service provider be held liable for violations of copyright or criminal law committed by its customers? Is Yahoo obliged to prevent French consumers from accessing a site where Nazi memorabilia is sold? Can copyright holders compel peer-to-peer file sharing systems to remove copyrighted material or to screen for such material? Are domain name registries required to check domain name registrations for trademark violations? Is eBay liable for counterfeit product sales on its site? To what extent should Google be allowed to offer excerpts from copyrighted books in its Google Book service without the consent of the relevant rights owners?

Both in the U.S. and in Europe, such questions have led to countless lawsuits and legislative initiatives over the last 15 years. One of the most debated issues in recent years has been whether Google is violating trademark law by operating its AdWords system. With Google AdWords, advertisers can buy advertising links in the "sponsored links" section of a Google search results page. When a user enters a keyword selected by the advertiser, the advertising link will appear in the upper right-hand corner of the search results page. In principle, the advertiser is free to select any keyword for his advertising link. This becomes a legal issue, however, if the advertiser chooses a

The European Court of Justice in Luxembourg.

PHOTOGRAPH BY GWENAËL PIASER

keyword that has been registered as a trademark by another company.

In 2003, the French fashion house Louis Vuitton discovered that, when French users entered "Louis Vuitton" into Google, they were shown an advertising link pointing to fake LV products. While LV could have sued the product imitator, it decided to sue Google. From LV's perspective, Google was a very attractive target: If Google was found liable, LV would not need to sue numerous individual product imitators. With one lawsuit against Google, LV could stop all keyword-related trademark violations at a stroke. Google, on the other hand, has a vital interest in avoiding being held liable in such lawsuits. Google's business model relies extensively on the advertisement auctioning mechanisms underlying the AdWords system. Of Google's $23.6 billion gross revenues in 2009, about $22.9 billion came from advertising (see http://investor.google.com/financial/tables.html). A major part of this advertising revenue is believed to come from Google AdWords.

Google's business model relies extensively on the advertisement auctioning mechanisms underlying the AdWords system.

Case Studies

Cases such as LV's have popped up like mushrooms over the last few years in many countries. From a trademark law perspective, they are not easy to resolve. On the one hand, it seems unfair that, by choosing third-party trademarks for keyword registrations without proper authorization, firms can benefit from the goodwill of such marks. It also seems problematic that Google may benefit, at least indirectly, from such behavior. On the other hand, trademark law does not protect trademark owners against each and every use of their registered marks by others. Where the Google AdWords system lies along this continuum is unclear.

In the French lawsuit of *Louis Vuitton vs. Google*, a Paris regional court found Google guilty of infringing LV's trademark in February 2005. After an appeals court in Paris had upheld this decision, Google appealed to the Cour de Cassation, which is the highest French court in this area of the law. The court had to decide whether or not Google AdWords was in compliance with French trademark law. At this point in the story the European Union kicks in. France has had a comprehensive trademark system since 1857. However, in 1989, the European Union required its member states to amend their national trademark systems in order to make them compliant with the European Trademark Directive enacted that year. This directive did not create a unitary Europewide trademark system. Rather, it harmonized national trademark systems across countries.[a] Today, if there is some disagreement about how a particular provision of national trademark law should be interpreted and whether this provision is affected by the European Trademark Directive, it is the European Court of Justice that has the last word. This was the case with the French LV litigation. As the highest court in France could not itself decide the case, in 2008, this court referred it to the European Court of Justice, which is located in Luxembourg.

The intellectual property community eagerly awaited the European Court of Justice's decision in this case. It was of particular importance because courts in various European countries had reached wildly different conclusions as to whether Google's AdWords system violates trademark law. Courts in France and Belgium, and some courts in Germany, had ruled that the AdWords system violates trademark law or unfair competition law, on the grounds that Google is using trademarks, confusing consumers, and free-riding on the goodwill of trademark owners. Courts in the U.K. and other courts in Germany have ruled the opposite, while decisions in Austria and the Netherlands have come out somewhere between these opposing viewpoints. Ultimately, in addition to the French Court de Cassation, the highest courts in Austria, Germany, the Netherlands, and the U.K. have referred AdWords-related lawsuits to the European Court of Justice.

In March 2010, the European Court of Justice decided the French LV case.[b] The court held that a producer of fake LV products violates trademark law if his keyword-backed advertising link creates the impression that his products are actually produced, or at least authorized by LV. This holding by the court was not surprising. More surprising was the court's holding that the fake product producer would violate trademark law even if he kept his advertisement so vague that ordinary consumers would be unable to determine whether or not there was some affiliation between the producer and LV. What this means in practice is unclear. While the European Court of Justice settled the relevant points of law, it did not provide a final answer as to whether the fake product producer was actually infringing trademark law. This depends on whether French consumers were really confused by the advertising link in question. As such matters of fact are not for the European Court of Justice to decide, the court referred the case back to the French courts in this regard.

The court then turned to the liability of Google itself. The court held that

a As a separate measure, the European Trademark Regulation of 1994 created a European-wide trademark system that is administered by the European trademark office (officially named the "Office of Harmonization for the Internal Market") in Alicante, Spain. As a result, two trademark systems now exist in Europe: the national trademark systems that are administered by national trademark offices and enforced by national courts, and the European trademark system that is administered by the European trademark office and is also enforced by national courts.

b This decision covered not only the lawsuit between LV and Google, but also two other related lawsuits in France, which will not be considered here. In addition, as of November 2010, the European Court of Justice has also ruled on AdWords-related cases from Austria, Germany, and the Netherlands. No decision on the U.K. case (Interflora) had been issued at the time of writing.

Google was not using the LV trademark in its AdWords system in a manner covered by European trademark law. The idea behind this is simple. Trademark law does not entitle a trademark owner to prevent all utilization of his trademark by a third party. In the view of the court, Google is merely operating a service that may enable advertisers to engage in trademark violations. Google does not decide which trademarks to use as keywords, but merely provides a keyword selection service. This is not sufficient, in the view of the court, to justify an action for direct trademark infringement.

However, Google might still be liable for what lawyers call secondary infringement. The argument would be that, if advertisers actually infringe trademark law because they create customer confusion in the AdWords system, Google is benefiting financially from these trademark violations. While this argument may sound convincing at first sight, the European E-Commerce Directive of 2000 restricts the liability of "information society service providers" (such as, potentially, Google) for infringing activities by third parties (the advertisers). Therefore, the European Court of Justice had to decide whether the safe harbor provisions of this directive shielded Google from secondary liability. The European Court of Justice held that the answer to this question depends on whether the Google AdWords system is a mere automatic and passive system, as portrayed by Google, or whether Google plays an active role in selecting and ordering advertisements. As in the customer confusion question, the court refrained from giving any definite answer, but rather referred the case back to the French courts.

In the popular press, the European Court of Justice's decision in the Google AdWords case has often been portrayed as a victory for Google. Does victory really look like this? Well, it depends. The European Court of Justice refrained from providing a final answer as to whether keyword advertising can lead to customer confusion. Nor did it provide a comprehensive answer as to whether Google could be held liable not because of customer confusion, but because other goals of trademark protection had been violated. Finally, the court did not give a definite answer as to whether Google should be protected by safe harbors provisions. For most of these questions, the European Court of Justice provided some general guidelines, but left it to the national courts to rule on details which may be small, but decisive. Therefore, in Europe, it will ultimately be the national courts which will decide on the liability of Google for its AdWords system. We still lack a clear answer on how to design a keyword-backed advertisement system in a way that clearly does not violate European trademark law.

The danger is that national courts will continue to interpret European trademark law in different ways.

Indecisive Decision

This does not mean that one should feel sorry for Google which still has to operate in an area of somewhat unsettled law. First, Google has some experience in this regard. Just think of the Google Books project. Second, Google has been running its AdWords service in the U.S. for years, and in the U.S. the liability question is still not fully settled. In 2009, the Court of Appeals for the Second Circuit held that Google was using trademarks "in commerce" (as required by the Lanham Act) when operating its AdWords system,[c] thereby taking a slightly different stance from that of the European Court of Justice. The impact of this decision on Google AdWords in the U.S. remains to be seen. At least, courts in the U.S. will now examine more closely whether unauthorized trademark-backed advertising links in Google AdWords can really cause confusion among consumers. Up to now, most U.S. courts have denied Google's liability on such grounds.

Third, as a result of the decisions by the European Court of Justice relating to the AdWords system, Google revised its European AdWords trademark policy in September 2010 and limited its support for trademark owners. Under the new policy, advertisers are free to select trademarks when registering advertising links. However, if a trademark owner discovers that an advertiser is using his trademark without proper authorization, Google will remove the advertising link if the trademark is being used in a confusing manner, for example if it falsely implies some affiliation between the advertiser and the trademark owner. By this policy change, Google has mollified at least some trademark owners and provided a mechanism outside the court system that may resolve a substantial proportion of AdWords trademark disputes in Europe. Nevertheless, it is almost certain that national courts in Europe will continue to rule on the details of how the AdWords trademark policy is implemented and enforced.

Conclusion

In the end, the decision by the European Court of Justice may indeed turn out to be a victory for Google. Whether it is a victory for the European trademark system is less clear. While the European Court of Justice provided some general guidelines on Google AdWords, the task of working out the little details has been left to courts in Paris, Vienna, Karlsruhe, The Hague, London and other cities. The danger is that national courts will continue to interpret European trademark law in different ways. French courts, for example, may continue to be more critical of Google AdWords in their decisions than German or U.K. courts. This is not exactly the idea of a trademark system which is supposed to be harmonized across Europe by the institutions of the European Union. ■

c *Rescuecom Corp. v. Google, Inc.*, 562 F.3d 123 (2009). This decision did not rule on the ultimate question of Google's liability, as the Court of Appeals remanded the case back to the district court for further proceedings. In March 2010, the parties settled their dispute out of court.

Stefan Bechtold (sbechtold@ethz.ch) is Associate Professor of Intellectual Property at ETH Zurich and a *Communications* Viewpoints section board member.

viewpoints

DOI:10.1145/1866739.1866750

Michael A. Cusumano

Technology Strategy and Management

Reflections on the Toyota Debacle

A look in the rearview mirror reveals system and process blind spots.

VARIOUS EXPERTS IN industry and academia have long recognized that Toyota, founded in 1936, is one of the finest manufacturing companies the world has ever seen.[a] Over the past 70-plus years, Toyota has evolved unique capabilities in manufacturing, quality control, supply-chain management, and product engineering, as well as sales and marketing. It began perfecting its famous Just-in-Time or "Lean" production system in 1948. I am a longtime observer (and customer) of Toyota, and have recently tried to understand how such a renowned company could experience the kinds of quality problems that generated numerous media headlines during 2009–2010.[b]

Example of an unsecured driver-side floor mat trapping the accelerator pedal in a 2007 Lexus ES350.

PHOTOGRAPH BY AP PHOTO/NHTSA

First, to recount some of the facts: Between 1999 and 2010, at least 2,262 Toyota vehicles sold in the U.S. experienced unintended cases of rapid acceleration and are associated with at least 815 accidents and perhaps as many as 102 deaths. The incidents that were not due to driver error (stepping on the gas pedal instead of the brake) appear to be the result of sticky brake pedals (easily fixed with a metal shim to replace a plastic component) as well as loose floor mats that inadvertently held down the gas pedal.[c] Another possible cause is the software that controls the engine and braking functions, particularly in vehicles such as the Prius and Lexus hybrids, which were also involved in the complaints. Toyota also encountered other quality problems that it mostly kept out of the headlines—in particular, dangerous corrosion in the frames of Tacoma and Tundra pickup trucks sold in North America between 1995 and 2000, apparently due to improper antirust treatment. Toyota did not recall these trucks, but silently bought them back from consumers.[d]

---

a See, for example, J. Womack et al., *The Machine that Changed the World* (1990); or J. Liker, *The Toyota Way* (2003).

b My first book, *The Japanese Automobile Industry* (1985), presented a history of how the Just-in-Time system was developed at Toyota. A later book, *Thinking Beyond Lean* (1998), examined Toyota's product development system. My most recent book, *Staying Power* (2010), looks back at Toyota's manufacturing, product development, and learning capabilities as well as how it ended up with these quality problems in 2009–2010.

c There are numerous reports on the Toyota problem in the media and information available from Toyota directly. A particularly detailed early document is *Toyota Sudden Unintended Acceleration*; www.safetyresearch.net. Also see "U.S. Safety Agency Reviewing More Crashes," *The Wall Street Journal*, (Feb. 15, 2010); http://online.wsj.com; and "Toyota's Sudden Acceleration Blamed for More Deaths," *Los Angeles Times* (Mar. 26, 2010); http://articles.latimes.com/2010/mar/26/business/la-fi-toyota-deaths26-2010mar26.

d Toyota's buyback program covered Tacoma pickup trucks made between 1995 and 2000. See "Toyota Announces Tacoma Buyback Program for Severe Rust Corrosion," *The Consumerist* (Apr. 15, 2008); http://consumerist.com/379734/toyota-announces-tacoma-buy-back-program-for-severe-rust-corrosion.

There have also been some minor complaints about the driving mechanisms in the Corolla and Camry models, and stalling in some Corolla models. Overall, during a 12-month period, Toyota recalled some 10 million vehicles through August 2010—an extraordinary number given that the company sold only approximately seven million vehicles during this same period.[2]

In the software business, producers and consumers are accustomed to product defects and an occasional recall as well as lots of "patches" or product fixes (see my earlier *Communications* column, "Who is Liable for Bugs and Security Flaws in Software?" March 2004, p. 25). Compared to automobiles, though, software product technology is relatively new, and the design and engineering processes are highly complex, especially for large systems with many interdependent components. But to what can we attribute so many quality problems in an industry as mature as automobiles and in a company so renowned for quality and manufacturing? Moreover, when even the mighty Toyota can falter, what does it say about "staying power"—the ability of a firm to sustain a competitive advantage and keep renewing or expanding its capabilities?

Systems and Managerial Process Problems

One way to think about the Toyota debacle is to divide the problem into categories: the production system, the product development system, and, for lack of a better term, general management. These systems and managerial processes also reflect intangible corporate values such as what kind of commitment the organization has to quality and customer satisfaction.

To what can we attribute so many quality problems in an industry as mature as automobiles and in a company so renowned for quality and manufacturing?

The Toyota production system does not seem to be the cause of the quality problems experienced over the prior decade. In the past, Toyota has exhibited a significant advantage over its mass-producer competitors in physical and value-added productivity. The competition has improved, but it is unlikely that any firm has actually passed Toyota in manufacturing prowess. Data related to manufacturing or assembly quality, such as the number of defects reported by customers in newly purchased vehicles, generally has placed Toyota at the top of the auto industry or at least among the leaders. This past year was different due to the recalls—Toyota fell from sixth to 21st in the annual J.D. Power's survey of initial quality.[3] However, the recent quality problems expose the limits of Toyota's production system. Making components or receiving supplier deliveries "just-in-time" as the assembly lines need the components minimizes inventory and operating costs, and exposes quality problems visible to assembly workers. But it does not detect design flaws that surface during usage of a product.

In terms of product development, including design and testing processes, Toyota has slipped a notch. The company seems to have tried too hard to reduce costs due to rising competition from low-cost but high-quality competitors such as Hyundai in Korea or new entrants in China. It is clearly a lapse in design and testing when accelerator pedals get stuck on loose floor mats, or when new types of plastic pedal materials become sticky after being exposed to moisture and friction. It is also a problem of design and testing when drivers feel that braking software or on-board computer controls and sensor devices seem to malfunction or operate crudely. Toyota's engineers and U.S. government safety investigators have not been able to replicate the conditions that caused some customers to complain about these software-related problems. But the kinds of problems we saw in 2009–2010 indicate Toyota engineers need to do a better job in product development to make sure their vehicles work properly under all conditions. Whether the component comes from an in-house Toyota factory or a supplier makes no difference. Toyota engineers are responsible.

In terms of general management, such as of the supply chain and overall quality, Toyota clearly failed to live up to its historical standards. Executives within the company admit they overstretched their managerial resources and overseas supply chain in the push to overtake General Motors as the world's largest automaker, which Toyota finally did in 2009. More specifically, the quality problems appear connected to overly rapid expansion of production and parts procurement outside Japan, particularly given the decision to use a different brake pedal. In the past, Toyota manufactured new models in Japan initially for a couple of years, using carefully tested Japanese parts, and only then did it move production of the best high-volume models to overseas factories. Over the last decade, by contrast, Toyota ramped up overseas production of new and old models with new suppliers much more quickly and, apparently, with inadequate stress testing.

Also at the management level, Toyota executives seem to have paid increasingly less attention to product and process details. It may well be that Toyota managers as well as staff engineers believed their company had already reached such a high level of perfection that there was nothing much to worry about. But automobiles are themselves very complex systems, with lots of hardware and software, and as many as 15,000 discrete components. It is not surprising that some things go wrong and recalls are common in the industry. Other automakers over the past year recalled more than 10 million vehicles, not counting the Toyota recalls.[2] In the grand scheme of things, moreover, the number of accidents and even deaths attributed to Toyota are not so large compared to what other companies have experienced. For example, Ford had a massive recall in 2000 of some 13 million faulty tires made by Firestone and fitted on its Explorer SUVs, reportedly resulting in over 250 deaths and 3,000 catastrophic injuries.[1,4] Nonetheless, Toyota redefined mass production and built its reputation around quality and reliability by paying attention to details, large and small. The recent slew of recalls definitely indicates something changed for the worse in the company.

Even the best firms are likely to decline at least a little as competitors catch up or when managers lose their focus.

What shocked me most was that the quality lapses seemed to take Toyota's senior managers by such surprise. CEO Akio Toyoda, and other senior executives in the U.S. and Japan, admitted to having little or no information about these quality issues, which first surfaced in Europe. They were unprepared to explain the source or nature of the problems—to themselves or to the global media. Toyota also made its predicament worse by responding much too slowly to customer complaints and allowing bad news to leak out sporadically, while executives continued to deny—at least initially—that there was a real problem.

Companies with true staying power fix their problems and recover from their mistakes. Here, Toyota has not disappointed us. By the fall of 2010, Toyota managers and dealers had gotten their act together and were working hard to rebuild customer confidence. The problems seemed mostly contained to the pedals and floor mats, though Toyota also upgraded some of the software in its hybrid vehicles. Service technicians worked overtime for months to fix recalled vehicles. Sales and profits recovered. And Toyota now recalls any vehicle immediately with even the slightest hint of a problem.

Technology and Management Lapses and Lessons

The Toyota debacle offers many lessons about technology and management. But one observation is that, although we can learn a lot about best practices from looking at exemplar firms and their unique processes, like Just-in-Time production, we also need to have some perspective. An enduring management principle that truly differentiates firms over the long haul must also be separable from the experience of any particular firm, including the originator. This sounds like a contradiction but it is not. Every company, market, and country will experience ups and downs. Even the best firms are likely to decline at least a little as competitors catch up or when managers lose their focus. Moreover, success often brings with it the potential seeds of decline—such as increases in the size, complexity, and global scale of operations, which can be much more difficult to manage. In this case, Toyota's quality problems in 2009–2010 do not mean the principles of "lean production" or lean management more generally are any less valuable to managers. What managers need to understand are the limitations of any best practice as well as the potential even for great companies to lose their focus and attention to detail—at least temporarily.

The best outcome for Toyota will be for managers, engineers, and other employees to reflect deeply on what happened to them and use these insights to create an even stronger company. They should become better able to handle adversity and change in the future because they now know what failure looks like. The Toyota way used to be that one defect was too many. That is the kind of thinking that Toyota seems to be regaining. ◼

References

1. Ackman, D. Tire trouble: The Ford-Firestone blowout. Forbes.com (June 20, 2001).
2. Bunkley, N. and Vlasic, B. Carmakers initiating more recalls voluntarily. *The New York Times* (Aug. 24, 2010); http://www.nytimes.com/2010/08/25/automobiles/25recall.html
3. Welch, W. Toyota plunges to 21st in auto-quality survey; Ford makes Top 5, Bloomberg (June 17, 2010); http://www.bloomberg.com/news/2010-06-17/toyota-plunges-to-21st-in-j-d-power-quality-survey-ford-makes-top-five.html.
4. Wikipedia. Firestone and Ford tire controversy. Wikipedia.com.

Michael A. Cusumano (cusumano@mit.edu) is a professor at the MIT Sloan School of Management and School of Engineering and author of *Staying Power: Six Enduring Principles for Managing Strategy and Innovation in an Uncertain World* (Oxford University Press, 2010).

DOI:10.1145/1866739.1866751

Mark D. Ryan

Viewpoint
Cloud Computing Privacy Concerns on Our Doorstep

Privacy and confidentiality issues in cloud-based conference management systems reflect more universal themes.

CLOUD COMPUTING MEANS entrusting data to information systems that are managed by external parties on remote servers "in the cloud." Webmail and online documents (such as Google Docs) are well-known examples. Cloud computing raises privacy and confidentiality concerns because the service provider necessarily has access to all the data, and could accidentally or deliberately disclose it or use it for unauthorized purposes.

Conference management systems based on cloud computing represent an example of these problems within the academic research community. It is an interesting example, because it is small and specific, making it easier to explore the exact nature of the privacy problem and to think about solutions. This column describes the problem, highlights some of the possible undesirable consequences, and points out directions for addressing it.

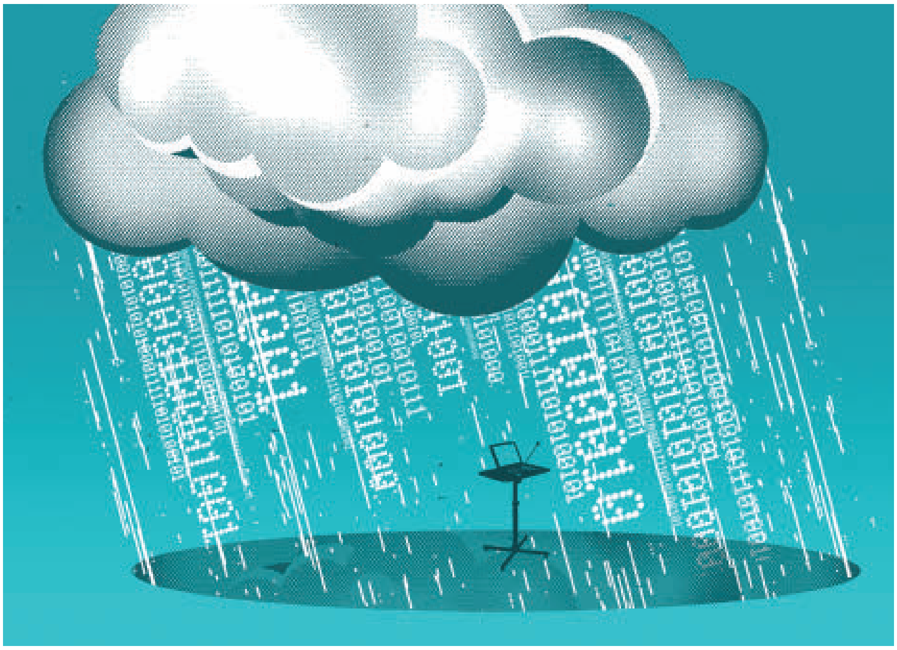


ILLUSTRATION BY GARY NEILL

Conference Management Systems

Most academic conferences are managed using software that allows the program committee (PC) members to browse papers and contribute reviews and discussion via the Web. In one arrangement, the conference chair downloads and hosts the appropriate server software, say HotCRP or iChair. The benefits of using such software are familiar:

- Distribution of papers to PC members is automated, and can take into account their preferences and conflicts of interest;
- The system organizes the collection and distribution of reviews and discussion, can rank papers according to scores, and send out reminder email, as well as email notifications of acceptance or rejection; and
- It can also produce a range of other reports, such as lists of sub-reviewers, acceptance statistics, and the conference program.

HotCRP and iChair require the conference chair to download and install software, and to host the Web server. Other systems such as EasyChair and EDAS work according to the cloud computing model: instead of installing and hosting the server, the conference chair simply creates the conference account "in the cloud." In addition to the benefits described previously, this model has extra conveniences:

- The whole business of managing the server (including backups and security) is done by someone else, and gains economy of scale;
- Accounts for authors and PC members exist already, and don't have to be managed on a per-conference basis;
- Data is stored indefinitely, and reviewers are spared the necessity of keeping copies of their own reviews; and

- The system can help complete forms such as the PC member invitation form and the paper submission form by suggesting likely colleagues based on past collaboration history.

For these reasons, EasyChair and EDAS are an immense contribution to the academic community. According to its Web page, EasyChair hosted over 3,300 conferences in 2010. Because of its optimizations for multiconferences and multitrack conferences, it is mandated for conferences and workshops that participate in the Federated Logic Conference (FLoC), a huge multiconference that attracts approximately 1,000 paper submissions.

Data Privacy Concerns

Accidental or deliberate disclosure. A privacy concern with cloud-computing-based conference management systems such as EDAS and EasyChair arises because the system administrators are custodians of a huge quantity of data about the submission and reviewing behavior of thousands of researchers, aggregated across multiple conferences. This data could be deliberately or accidentally disclosed, with unwelcome consequences.

- Reviewer anonymity could be compromised, as well as the confidentiality of PC discussions.
- The acceptance success records could be identified, for individual researchers and groups, over a period of years; and
- The aggregated reviewing profile (fair/unfair, thorough/scant, harsh/undiscerning, prompt/late, and so forth) of researchers could be disclosed.

The data could be abused by hiring or promotions committees, funding and award committees, and more generally by researchers choosing collaborators and associates. The mere existence of the data makes the system administrators vulnerable to bribery, coercion, and/or cracking attempts. If the administrators are also researchers, the data potentially puts them in situations of conflict of interest.

The problem of data privacy in general is of course well known, but cloud computing magnifies it. Conference data is an example in our backyard. When conference organizers had to install the software from scratch, there was still a risk of breach of confidentiality, but the data was just about one conference. Cloud computing solutions allow data to be aggregated across thousands of conferences over decades, presenting tremendous opportunities for abuse if the data gets into the wrong hands.

The acceptance success records could be identified, for individual researchers and groups, over a period of years.

Beneficial data mining. In addition to the abuses of conference review data described here, there are some uses that might be considered beneficial. The data could be used to help detect or prevent fraud or other kinds of unwanted behavior, for example, by identifying:

- Researchers who systematically unfairly accept each other's papers, or rivals who systematically reject each other's papers, or reviewers who reject a paper and later submit to another conference a paper with similar ideas; and
- Undesirable submission patterns and behaviors by individual researchers (such as parallel or serial submissions of the same paper; repeated paper withdrawals after acceptance; and recurring content changes between submitted version and final version).

The data could also be used to understand and improve the way conferences are administered. ACM, for example, could use the data to construct quality metrics for its conferences, enabling it to profile the kinds of authors who submit, how much "new blood" is entering the community, and how that changes over different editions of the conference. This could help identify conferences that are emerging as dominant, or others that have outlived their usefulness.

The decisions about who is allowed to mine the data, and for what purposes, are difficult. Policies should be decided transparently and by consensus, rather than being left solely to the de facto data custodians.

Ways Forward

Policies and legislation. An obvious first step is to articulate clear policies that circumscribe the ways in which the data is used. For example, a simple policy might be that the data gathered during the administration of a conference should be used only for the management of that particular conference. Adherence to this policy would imply that the data is deleted after the conference, which is not done in the case of Easychair (I don't know if it is done for EDAS). Other policies might allow wider uses of the data. Debate within different academic communities can be expected to yield consensus about which practices are to be allowed in a discipline, and which ones not. For example, some communities may welcome plagiarism detection based on previously reviewed submissions, while others may consider it useless for their subject, or simply unnecessary.

Since its inception in 2002 and up to the time of writing, EasyChair has appeared not to have any privacy policy, or any statement about the purposes and possible uses of the data it stores. There is no privacy policy linked from its main page, and a search for "privacy policy" (or similar terms) restricted to the domain "easychair.org" does not yield any results. I have been told that new users are presented with a privacy statement at the time of first signing up to Easychair. I did not create a new account to test this; regardless, the privacy statement is not linked from anywhere or later findable via search. EDAS does have an easily accessed privacy policy, which (while not watertight) appears to comply with the "use only for this conference" principle.

Another direction would be to try to find alternative custodians for the data—custodians that are not themselves also researchers participating actively in conferences. The ACM or IEEE might be considered suitable, although they contribute to decisions about publications and appointments of staff and fellows. Professional data custodians such as Google might also be considered. It may be difficult to find an ideal custodian, especially if cost factors are taken into account.

In most countries, legislation exists to govern the protection of personal data. In the U.K., the Data Protection Act is based on eight principles, including the principle that personal data is obtained only for specified purposes and is not processed in a manner incompatible with the purposes; and the principle that the data is not kept longer than is necessary for the purposes. EasyChair is hosted in the U.K., but the lack of an accessible purpose statement or evidence of registration under the Act mean I was unable to determine whether it complies with the legislation. The Data Protection Directive of the European Union embodies similar principles; personal data can only be processed for specified purposes and may not be processed further in a way incompatible with those purposes.

Processing encrypted data in the cloud. Policies are a first step, but alone they are insufficient to prevent cloud service providers from abusing the data entrusted to them. Current research aims to develop technologies that can give users guarantees that the agreed policies are adhered to. The following descriptions of research directions are not exhaustive or complete.

Progress has been made in encryption systems that would allow users to upload encrypted data, and allow the service providers to perform computations and searches on the encrypted data without giving them the possibility of decrypting it. Although such encryption has been shown possible in principle, current techniques are very expensive in both computation and bandwidth, and show little sign of becoming practical. But the research is ongoing, and there are developments all the time.

Hardware-based security initiatives such as the Trusted Platform Module and Intel's Trusted Execution Technology are designed to allow a remote user to have confidence that data submitted to a platform is processed according to an agreed policy. These technologies could be leveraged to give privacy guarantees in cloud computing in general, and conference management software in particular. However, significant research will be needed before a usable system could be developed.

Certain cloud computing applications may be primarily storage applications, and might not require a great deal of processing to be performed on the server side. In that case, encrypting the data before sending it to the cloud may be realistic. It would require keys to be managed and shared among users in a practical and efficient way, and the necessary computations to be done in a browser plug-in. It is worthwhile to investigate whether this arrangement could work for conference management software.

Conclusion

Many people with whom I have discussed these issues have argued that the professional honor of data custodians (and PC chairs and PC members) is sufficient to guard against the threats I have described. Indeed, adherence by professionals to ethical behavior is essential to ensure all kinds of confidentiality. In practice, system administrators are able to read all the organization's email, and medical staff can browse celebrity health records; we trust our colleagues' sense of honor to ensure these bad things don't happen. But my standpoint is that we should still try to minimize the extent to which we rely on people's sense of good behavior. We are just at the beginning of the digital era, and many of the solutions we currently accept won't be considered adequate in the long term.

The issues raised about cloud-computing-based conference management systems are replicated in numerous other domains, across all sectors of industry and academia. The problem of accumulations of data on servers is very difficult to solve in any generality. The particular instance considered here is interesting because it may be small enough to be solvable, and it is also within the control of the academic community that will directly benefit—or suffer—according to the solution we adopt. C

Mark D. Ryan (M.D.Ryan@cs.bham.ac.uk) is Professor in Computer Security and EPSRC Leadership Fellow in the School of Computer Science at the University of Birmingham, U.K.

Many thanks to the *Communications* reviewers for interesting and constructive comments. I also benefited from discussions with many colleagues at Birmingham, and also in the wider academic research community. Thanks to Henning Schulzrinne, administrator of EDAS, for comments and clarifications. Drafts of this Viewpoint were sent to Andrei Voronkov, the Easychair administrator, but he did not respond.

DOI:10.1145/1866739.1866752

Guy L. Steele Jr.

Interview

An Interview with Frances E. Allen

Frances E. Allen, recipient of the 2006 ACM A.M. Turing Award, reflects on her career.

ACM FELLOW FRANCES E. ALLEN, recipient of the 2006 ACM A.M. Turing Award and IBM Fellow Emerita, has made fundamental contributions to the theory and practice of program optimization and compiler construction over a 50-year career. Her contributions also greatly extended earlier work in automatic program parallelization, which enables programs to use multiple processors simultaneously in order to obtain faster results. These techniques made it possible to achieve high performance from computers while programming them in languages suitable to applications. She joined IBM in 1957 and worked on a long series of innovative projects that included the IBM 7030 (Stretch) and its code-breaking coprocessor Harvest, the IBM Advanced Computing System, and the PTRAN (Parallel Translation) project. She is an IEEE Fellow, a Fellow of the Computer History Museum, a member of the American Academy of Arts and Sciences, and a member of the U.S. National Academy of Engineering.

ACM Fellow Guy L. Steele Jr. visited Allen at her office in the IBM T.J. Watson Research Center in 2008 for an extended oral interview. The complete transcript of this interview is available in the ACM Digital Library; presented here is a condensed version that highlights Allen's technical accomplishments and provides some anecdotes about her colleagues.

Fran Allen on CS: "It's just such an amazing field, and it's changed the world, and we're just at the beginning..."

PHOTOGRAPH BY FRANK BECERRA, JR. / THE JOURNAL NEWS

Your first compiler work was for IBM Stretch.[a]

Yes. In 1955, IBM recognized that to be 100 times faster than any machine existing or planned at the time, the major performance problem to overcome was latency to memory. The advanced ideas and technologies they put into Stretch were to address that problem. Six instructions could be in flight at the same time. The internal memory was interleaved, and data would arrive out of order—data and instructions were both stored in this memory. They built a very complex buffering system and look-ahead. John Cocke, when he came in 1956, was put in charge of the look-ahead for instructions. It was also architected to have precise interrupts. So

a See the December 2010 *Communications* Historical Perspectives column "IBM's Single-Processor Supercomputer Efforts" for more discussion of the IBM Stretch supercomputer.

the look-ahead unit was a phenomenal piece of hardware.

How many copies of Stretch were built?

Eight or nine. The original was built for Los Alamos and shipped late. Then they discovered its performance was about half of what was intended.

But still, 50 times...

Meanwhile, the underlying technology had changed. T.J. Watson got up at the Spring Joint Computer Conference and announced they would not build any more Stretch machines, and apologized to the world about our failure. But it was recognized later that the technology developed in building Stretch made a huge difference for subsequent machines, particularly the 360. A lot of people went from Stretch to the 360, including Fred Brooks.

What was your connection with Stretch?

My role was on the compiler. When I joined IBM in 1957, I had a master's degree in mathematics from the University of Michigan, where I had gone to get a teaching certificate to teach high school math. But I had worked on an IBM 650 there, so I was hired by IBM Research as a programmer. My first assignment was to teach FORTRAN, which had come out in the spring of that year.

Did you already know FORTRAN, or were you learning it a week ahead, as professors often do?

Yeah, a week ahead [laughs]. They had to get their scientists and researchers to use it if they were going to convince

It was recognized later that the technology developed in building Stretch made a huge difference for subsequent machines.

customers. I had a pretty unhappy class, because they knew they could do better than any high-level language could.

Did you win them over?

Yes—and won myself over. John Backus, who led the FORTRAN project, had set two goals from the beginning: programmer productivity and application performance. I learned all about the compiler as part of teaching this course.

Did you ever work on that compiler yourself?

I was reading the code in order to do the training. It set the way I thought about compilers. It had a parser, then an optimizer, then a register allocator. The optimizer identified loops, and they built control flow graphs.

The Stretch group recognized that the compiler was going to be an essential part of that system. A bunch of us in research were drafted to work on it. The National Security Agency [NSA] had a contract with IBM to build an add-on to Stretch, for code-breaking. Stretch would host the code-breaking component, and there was a large tape device, tractor tape, for holding massive amounts of data.

This was Stretch Harvest?

Yes. There was going to be one compiler for Stretch Harvest that would take FORTRAN, and the language I was working on with NSA for code-breaking, called Alpha, and also Autocoder, which was similar to COBOL.

A single compiler framework to encompass all three languages?

Yes, three parsers going to a high-level intermediate language, then an optimizer, then the register allocator. It was an extraordinarily ambitious compiler for the time, when even hash tables were not yet well understood. One compiler, three source languages, targeted to two machines, Stretch and Harvest. In addition to managing the optimizer group, I was responsible for working with the NSA on designing Alpha. I was the bridge between the NSA team, which knew the problem...

And never wanted to tell you completely what the problem is.

They told me not at all, but it didn't matter. I was pretty clueless about everything down there, and on purpose. "NSA" was not a term that was known. While I was on the project, two guys went to Moscow, just left, and it hit the *New York Times*, and that's when I learned what it was about. It was a very carefully guarded activity. The problem was basically searching for identifiers in vast streams of data and looking for relationships, identifying *k*-graphs and doing statistical analysis. Any single Harvest instruction could run for days, and be self-modifying.

The most amazing thing about that machine is that it was synchronized. Data flowed from this tape system through memory, through the streaming unit, to the Harvest unit, the streaming unit, back to memory, and back out onto the data repository, and it was synchronized at the clock level.

The data was coming from listening stations around the world, during the Cold War. I spent a year at NSA installing the system; during that year, the Bay of Pigs and the Cuban Missile Crisis happened, so it was a very tense period. I assume most of the data was in Cyrillic. But Alpha could deal with any data that had been coded into bytes.

I wrote the final acceptance test for the compiler and the language. I wrote the final report and gave it to them and never saw it again, which I regret.

What did you do next?

John Cocke was enamored with building the fastest machine in the world, and Stretch had been an announced public failure. When I finished with Harvest, Stretch was already done. I could have gone and worked on the 360. I didn't particularly want to do that; it was a huge project spread around the world. John wanted to take another crack at building the fastest machine in the world, so I joined him on a project called System Y. This time the compiler was built first. Dick Goldberg was the manager and did the parser, I did the optimizer, and Jim Beatty did the register allocator. We had a very nice cycle-level timing simulator. We built what was called the Experimental Compiling System.

What became of System Y?

It changed into ACS [Advanced Computing System], which was eventually canceled [in 1969] by Armonk, by headquarters, which we should have known would happen, because it was not 360. But we developed things that subsequently influenced the company a lot. We did a lot with branch prediction, both hardware and software, and caching, and machine-independent, language-independent optimizers. John, after being very disappointed about not being able to build the fastest machine in the world, decided he would build the best cost-performance machine. That was where the PowerPC came from—the 801 project.

After ACS, I took an unhappy digression from my work on compilers. I was assigned to work on FS, the famous "Future System" of IBM. It was so bad on performance, I wrote a letter. FS took two round trips to memory to fetch any item of data, because it had a very high-level intermediate form as the architected form for the machine.

Should I be reminded of the Intel 432, the processor designed for Ada? It had a very high-level architecture that turned out to be memory-bound, because it was constantly fetching descriptors from memory.

Yes. We aren't very good about passing on the lessons we've learned, and we don't write our failures up very well.

It's harder to get a failure published than a success.

But there are a lot of lessons in them. After fuming about FS for a few months, I wrote a letter to somebody higher up and said, "This isn't going to work," and why, and that was the wrong thing to say. So I was kind of put on the shelf for a while. But then I did a lot of work with a PL/I compiler that IBM had subcontracted to Intermetrics.

The compilers you worked on—such as the ACS compiler and the PL/I compiler in the 1970s—what languages were those implemented in?

Some of them were implemented in FORTRAN, some in PL/I, and some were in assembly language.

How about Alpha?

That was in the assembly language for Stretch.

Your 1976 *Communications* paper with John Cocke contains some actual PL/I code that represents sets as bit vectors, and propagates sets around the program control flow graph. The intersections and unions of the sets were just PL/I & and | operators, which makes the code concise and easy to read. You have said that PL/I was a complicated language to compile, but it seems to have expressive power.

> **Any single Harvest instruction could run for days, and be self-modifying.**

Yes, it was really very useful for writing optimizers and compilers. The data flow work came from early FORTRAN and their use of control flow graphs. On Project Y we built control flow graphs and developed a language about the articulation points on the graph, abstracting away from DO loops into something more general, then optimizing based on a hierarchy of these graphs, making the assumption that they represented parts of the program that were most frequently executed.

When did you first start using that bit-vector representation?

Right at the beginning of the ACS project. "Graph intervals" was a term that John had come up with, but then I wrote the paper and carried the idea further. Then Mike Harrison came, and we were struggling with the problem that we had no way of bounding the computation of the flow of information in such a graph.

In some of your papers, you talked about earlier monotonic relaxation techniques, but they had very large theoretical bounds.

Yes, but I wasn't much concerned, because I knew that real programs don't have those, and Mike agreed. Jeff Ullman did some analysis on programs. That did get a better bound, but that analysis didn't produce a structure against which one could actually make transformations.

The graph interval decomposition improved the theoretical cost bounds of the algorithm by guiding the order—but if I hear you correctly, the interval structure is just as important, perhaps more important, for guiding the transformations than for doing the analysis?

Yes. People who were focusing on the theoretical bounds missed, I think, the importance of leaving a framework in which one could make the transformations. But then something really exciting happened. A student of Knuth's, [Robert] Tarjan, developed a way to map this problem into a spanning tree.

Nodal graphs could be decomposed into spanning trees plus back edges.

Yes! It was startling. Great things sometimes look simple in retrospect, but that solved that part of structuring the bounds of subsequent algorithms' analysis and transformation.

So Tarjan's work played a role in this?

Yes, I don't think he knew it, but as soon as he published that, it was just obvious that we should abandon graph intervals and go there.

Could you talk about Jack Schwartz?

Jack spent a summer at ACS and had a huge influence. He wrote a number of wonderful papers on optimizing transformations, one being "Strength reduction, or Babbage's differencing engine in modern dress." Jack had a list of applications for strength reduction, which we in compilers never took advantage of. He and John wrote a big book, never published but widely circulated, on a lot of this work. I spent a year in the Courant Institute—I taught graduate compilers. And Jack and I were married for a number of years. So it was a good relationship all around.

What did you think about SETL [a programming language developed by Schwartz]?

It wasn't the right thing for that time, but it may be an interesting language to go back and look at now that we're mired in over-specifying.

Gregory Chaitin's classic PLDI paper on "Register Allocation and Spilling via Graph Coloring" contains a substan-

tial chunk of SETL code, four and a half pages, that implements the algorithm.

I liked SETL and was amazed that they got some good compiling applications out of it. In the context of multicores and all the new challenges that we've got, I like it a lot—it's one instance of specifying the problem at such a high level that there's a good possibility of being able to target multiple machines and to get high performance from programs that are easy to write.

I have a story about register allocation. FORTRAN back in the 1950s had the beginnings of a theory of register allocation, even though there were only three registers on the target machine. Quite a bit later, John Backus became interested in applying graph coloring to allocating registers; he worked for about 10 years on that problem and just couldn't solve it. I considered it the biggest outstanding problem in optimizing compilers for a long time. Optimizing transformations would produce code with symbolic registers; the issue was then to map symbolic registers to real machine registers, of which there was a limited set. For high-performance computing, register allocation often conflicts with instruction scheduling. There wasn't a good algorithm until the Chaitin algorithm. Chaitin was working on the PL.8 compiler for the 801 system. Ashok Chandra, another student of Knuth's, joined the department and told about how he had worked on the graph coloring problem, which Knuth had given out in class, and had solved it—not by solving the coloring problem directly, but in terms of what is the minimal number of colors needed to color the graph.

> **The Stretch look-ahead was designed on bar napkins, particularly in the Old Brauhaus in Poughkeepsie.**

Greg immediately recognized that he could apply this solution to the register allocator issue. It was a wonderful kind of serendipity.

Anything else we should know about John Cocke?

He had a major impact on everybody. Let me talk about his style of work. He didn't write anything, and giving a talk was exceedingly rare and painful for him. He would walk around the building, working on multiple things at the same time, and furthered his ideas by talking to people. He never sat in his office—he lost his tennis racket one time for several months and eventually found it on his desk. If he came into your office, he would start drawing and pick up the conversation exactly where he had left off with you two weeks ago!

So he was very good at co-routining!

Yes, he could look at a person and remember exactly the last thing he said to them. And people used to save his bar napkins. He spent a lot of time in bars; he liked beer. He would draw complex designs on napkins, and people would take the napkins away at the end of the evening. The Stretch look-ahead was designed on bar napkins, particularly in the Old Brauhaus in Poughkeepsie.

You also knew Andrei Ershov.

He did some marvelous work in the Soviet Union. Beta was his compiler, a really wonderful optimizing compiler. He had been on the ALGOL committee.

He had an earlier project that he called Alpha, not to be confused with the Alpha language you did for Stretch, right?

No, it was totally unrelated. But later we read his papers. Then in 1972 he couldn't travel, because he wasn't a party member, so he had a workshop in Novosibirsk and invited a large number of people. It was broader than compilers, but there was a big focus on compilers, and we picked up some things from his work.

Ershov also worked with people in China. When the curtain came down between the Soviet Union and China, the Chinese group then didn't have access to Ershov's work. Jack and I were invited to China in 1973 to lecture. Mao was still alive, and a lot of the institutes and universities were pretty much closed. There was a science institute in Peking and in Shanghai, where we gave talks on compilers, and we looked at the machines there, which were really quite primitive. The compiler they were running on the machine in Peking was on paper tape. I recognized, looking at the code, that it was essentially Ershov's compiler. So the people in China were really quite concerned about being cut out of the advances in computing. This is a conjecture I've only recently arrived at, why we in particular in the U.S. were asked to come: it was a connection through the technology that the three groups shared. We were very involved with Ershov and his group. He and his family wanted to leave the Soviet Union, and they lived with us in our home for about a year.

You actually had two projects called "Experimental Compiling System." What was the second one like?

Its overall goals were to take our work on analysis and transformation of codes, and embed that knowledge in a schema that would advance compiling. I wish we had done it on Pascal or something like that.

PL/I was that difficult a language?

Yes, it was the pointers and the condition handling—those were the big problems. This was another bold project, and my interest was mostly in the generalized solution for interprocedural analysis—but also putting what we knew into a context that would make writing compilers easy and more formal, put more structure into the development of compilers. We already had a lot of great algorithms which we could package up, but this was to build a compiler *framework* where the methods that we already had could be used more flexibly.

Did lessons learned from this project feed forward into your PTRAN work?

The interprocedural work did, absolutely, and to some extent the work on binding. It sounds trivial, but constant propagation, getting that right, and being able to take what you know and refine the program without having to throw things away and start over.

Let's talk about PTRAN. Two papers came out in 1988: your "Overview of the PTRAN Analysis System" and "IBM Parallel FORTRAN". It's important to distinguish these two projects. IBM Parallel FORTRAN was a product, a FORTRAN augmented with constructs such as PARALLEL LOOP and PARALLEL CASE and ORIGINATE TASK. So the FORTRAN product is FORTRAN with extra statements of various kinds, whereas with PTRAN, you were working with raw FORTRAN and doing the analysis to get parallelism.

Right.

What was the relationship between the two projects? The IBM Parallel FORTRAN paper cites your group as having provided some discussion.

The PTRAN group was formed in the early 1980s, to look first at automatic vectorization. IBM was very late in getting into parallelism. The machines had concurrency, but getting into explicit parallelization, the first step was vectorization of programs. I was asked to form a compiler group to do parallel work, and I knew of David Kuck's work, which started in the late 1960s at the University of Illinois around the ILLIAC project. I visited Kuck and hired some of his students. Kuck and I had a very good arrangement over the years. He set up his own company—KAI.

Kuck and Associates, Inc.

Right. IBM, at one point later on, had them subcontracted to do some of the parallelism. They were very open about their techniques, with one exception, and they were the leaders early on. They had a system called Parafrase, which enabled students to try various kinds of parallelizing code with FORTRAN input and then hooked to a timing simulator back-end. So they could get real results of how effective a particular set of transformations would be. It was marvelous for learning how to do parallelism, what worked and what didn't work, and a whole set of great students came out of that program. In setting up my group, I mostly hired from Illinois and NYU. The NYU people were involved with the Ultracomputer, and we had a variant of it here, a project called RP3, Research Parallel Processor Prototype, which was an instantiation of their Ultracomputer.

Another thing I think was a very big step was not only identifying parallelism, but identifying *useful* parallelism.

The Ultracomputer was perhaps the first to champion fetch-and-add as a synchronization primitive.

Yes. A little history: The Ultracomputer had 256 processors, with shared distributed memory, accessible through an elaborate switching system. Getting data from memory is costly, so they had a combining switch, one of the big inventions that the NYU people had developed. The fetch-and-add primitive could be done in the switch itself.

Doing fetch-and-add in the switch helped avoid the hot-spot problem of having many processors go for a single shared counter. Very clever idea.

Very, very clever. So IBM and NYU together were partners, and supported by DARPA to build a smaller machine. The number of processors got cut back to 64 and the combining switch was no longer needed, and the project kind of dragged on. But my group supplied the compiler for that. The project eventually got canceled.

So that was the background, in IBM Research and at the Courant Institute. But then the main server line, the 370s, 3090s, were going to have vector processors.

Multiple vector processors as well as multiple scalar processors.

Yes. And the one that we initially worked on was a six-way vector processor. We launched a parallel translation group, PTRAN. Jean Ferrante played a key role. Michael Burke was involved; NYU guy. Ron Cytron was the Illinois guy. Wilson Hsieh was a co-op student. Vivek Sarkar was from Stanford, Dave Shields and Philippe Charles from NYU. All of these people have gone on to have some really wonderful careers. Mark Wegman and Kenny Zadeck were not in the PTRAN group but were doing related work. We focused on taking dusty decks and producing good parallel code for the machines—continuing the theme of language-independent, machine-independent, and do it automatically.

"Dusty decks" refers to old programs punched on decks of Hollerith cards. Nowadays we've got students who have never seen a punched card.

We also went a long way with working with product groups. There was a marvelous and very insightful programmer, Randy Scarborough, who worked in our Palo Alto lab at the time. He was able to take the existing FORTRAN compiler and add a little bit or a piece into the optimizer that could do pretty much everything that we could do. It didn't have the future that we were hoping to achieve in terms of building a base for extending the work and applying it to other situations, but it certainly solved the immediate problem very inexpensively and well at the time. That really helped IBM quickly move into the marketplace with a very parallel system that was familiar to the customers and solved the problem. Disappointing for us, but it was the right thing to have happen.

Did PTRAN survive the introduction of this product?

Yes, it survived. The product just did automatic vectorization. What we were looking at was more parallelism in general.

One particular thing in PTRAN was looking at the data distribution problem, because, as you remarked in your paper, the very data layouts that improve sequential execution can actually harm parallel execution, because you get cache conflicts and things like that.

Yes.

That doesn't seem to be addressed at all by the "IBM Parallel FORTRAN" paper. What kinds of analysis were you doing in PTRAN? What issues were you studying?

Well, two things about the project.

We worked on *both* the theory and abstraction, building up methods that were analyzable and could be reasoned about, *and* implementing them. I insisted in this project that, if someone on the systems side showed me a piece of code, I would say, "Can you describe this in a paper? How would other people know about this?" If they were on the theoretical side, I would say, "Can you implement it? Show me the implementation."

The idea of trying to change a program into a functional program was something that I was trying to push. We could do much better analysis even for just plain straight optimization if we could name the values but not burden it with the location, apply a functional paradigm to it.

We could trace that idea back to your early work on strength reduction, when you were making hash names for intermediate values.

Yes. The value contributes to the answer, but where that value resides should be irrelevant to the writer of the program.

Apparently, just convincing the early programmers of that was one of your early successes. FORTRAN is good enough; you don't need to keep track of every single machine register yourself.

That's right. So I had that challenge out there. We needed to try and recast the program as close as we could to functional.

Another thing I think was a very big step was not only identifying parallelism, but identifying *useful* parallelism.

Another problem: say that one of the optimizations is constant propagation. For some variable deep in the code,

That was a problem we struggled with early on: How do you avoid redoing the analysis?

there is a constant that you have recognized could replace the use of the variable. You then know, say, which way a branch is going to go. You've built up this infrastructure of analysis and you're ready to make the transformation—but then the results of the analysis are obsolete, so you have to start again. That was a problem we struggled with early on: How do you avoid redoing the analysis? It got particularly bad with interprocedural activities.

Is there some simple insight or overarching idea that helps you to avoid having to completely redo the computation?

Vivek Sarkar was one of the key people on that, but Dave Kuck—this is at the core of KAI's work, too. That group described it as "the oracle." You assign costs to each of the instructions, and you can do it in a hierarchical form, so this block gets this cost, and this block has that cost, and then do a cost analysis. This is the time it's going to take. Then there's the overhead cost of having the parallelism.

Earlier, you said that Kuck was very open about everything he was doing, with one exception—

The oracle! "What have you got in that thing?" [laughs] "We're not going to tell you!" So we built our own variant of it, which was a very powerful technique.

What else should we mention?

We talked about the NSA work that wasn't published. That was, for me, a mind-changer that led to my feeling very strongly about domain-specific languages.

Are you for them or against them?

For them!

Oh, okay. Let's be clear! [laughs]

Good! [laughs]

I'm going to try something very foolish: summarize your career in one paragraph, then ask you to critique it. *A major focus of your career has been that, rather than inventing new programming languages or language features and trying to get people to program in them, you focused on taking programs as they are written, or as programmers like to write them, and making them run really effectively and efficiently on target machines. One of the many ways you do this, but a very important one, is to do many kinds of sophisticated analysis and optimization of the code and to find out as much as you can about the characteristics of the program without actually running it. So these tend to be static techniques, and very sophisticated ones. While you have worked with and pioneered quite a number of them, some of the most interesting involve using graphs as a representation medium for the program and using a strategy of propagating information around the graph. Because a program can be represented as a graph in more than one way, there's more than one way in which to propagate that information. In some of these algorithms in particular, the information that's being propagated around the graph is in the form of sets—for example, sets of variable names. As a strategy for making some of these algorithms efficient enough to use, you've represented sets as bit vectors and decomposed the graphs using interval analysis in order to provide an effective order in which to process the nodes. In doing this, you have built a substantial sequence of working systems; these aren't just paper designs. You build a great system, and then you go on and build the next one, and so on. These all actually work on code and take real programs that aren't artificial benchmarks and make them run.*

That's really very good. There's one thing: the overall goal of all of my work has been the FORTRAN goal, John Backus' goal: user productivity, application performance.

Now, three goofy questions. What's your favorite language to compile?

FORTRAN, of course!

What's your favorite language to program *in*?

I guess it would have to be FORTRAN.

Okay, now, if you had to build a compiler that would run on a parallel machine, what language would you use to write that compiler?

Probably something like SETL or a functional language. And I'm very intrigued about ZPL. I really liked that language.

Any advice for the future?

Yes, I do have one thing. Students aren't joining our field, computer science, and I don't know why. It's just such an amazing field, and it's changed the world, and we're just at the beginning of the change. We have to find a way to get our excitement out to be more publicly visible. It *is* exciting—in the 50 years that I've been involved, the change has been astounding. ■

Recommended Reading

Buchholz, W., Ed.
Planning a Computer System: Project Stretch. **McGraw-Hill, 1962; http://ed-thelen.org/comp-hist/IBM-7030-Planning-McJones.pdf**

Allen, F.E. and Cocke, J.
A catalogue of optimizing tranformations. In R. Rustin, Ed., *Design and Optimization of Compilers*. Prentice-Hall, 1972, 1–30.

Allen, F.E.
Interprocedural data flow analysis. In *Proceedings of Information Processing 74*. IFIP. Elsevier/North-Holland, 1974, 398–402.

Allen, F.E. and Cocke, J.
A program data flow analysis procedure. *Commun. ACM 19*, 3 (Mar. 1976), 137–147; http://doi.acm.org/10.1145/360018.360025

Allen, F.E. et al.
The Experimental Compiling System. *IBM J. Res. Dev. 24*, 6 (Nov. 1980), 695–715.

Allen, F.E.
The history of language processor technology at IBM. *IBM J. Res. Dev. 25*, 5 (Sept. 1981), 535–548.

Allen, F.E. et al.
An overview of the PTRAN analysis system for multiprocessing. In *Proceedings of the 1st International Conference on Supercomputing* (Athens, Greece, 1988), Springer-Verlag, 194–211. Also in J. Par. Dist. Comp. 5 (Academic Press, 1988), 617–640.

Recommended Viewing

Allen, F.E.
The Stretch Harvest compiler. Computer History Museum, Nov. 8, 2000. Video, TRT 01:12:17; http://www.computerhistory.org/collections/accession/102621818

The IBM ACS System: A Pioneering Supercomputer Project of the 1960s. Speakers: Russ Robelen, Bill Moone, John Zasio, Fran Allen, Lynn Conway, Brian Randell. Computer History Museum, Feb. 18, 2010; Video, TRT 1:33:35; http://www.youtube.com/watch?v=pod53_F6urQ

DOI:10.1145/1866739.1866755

Article development led by acmqueue
queue.acm.org

For sysadmins, solving problems usually involves collaborating with others. How can we make it more effective?

BY EBEN M. HABER, ESER KANDOGAN, AND PAUL P. MAGLIO

Collaboration in System Administration

GEORGE WAS IN trouble. A seemingly simple deployment was taking all morning, and there seemed no end in sight. His manager kept coming in to check on his progress, as the customer was anxious to have the deployment done. He was supposed to be leaving for a goodbye lunch for a departing co-worker, adding to the stress. He had called in all kinds of help, including colleagues, an application architect, technical support, and even one of the system developers. He used email, instant messaging, face-to-face contacts, his phone, and even his office mate's phone to communicate with everyone. And George was no novice. He had been working as a Web-hosting administrator for three years, and he had a bachelor's degree in computer science. But it seemed that all the expertise being brought to bear was simply not enough. Why was George in trouble? We'll find out.

But first, why were we watching George? George is a system administrator, one of the people who work behind the scenes to configure, operate, maintain, and troubleshoot the computer infrastructure that supports much of modern life. Their work is critical—and expensive. The human part of total system cost-of-ownership has been growing for decades, now dominating the costs of hardware or software.[2–4]

To understand why, and to try to learn how administration can be better supported, we have been watching system administrators at work in their natural environments. Over the course of several years, and equipped with camcorders, cameras, tapes, computers, and notebooks, we made 16 visits, each as long as a week, across six different sites. We observed administrators managing databases, Web applications, and system security; as well as storage designers, infrastructure architects, and system operators. Whatever their specific titles were, we refer to them all as system administrators, or sysadmins for short.

At the beginning of our studies, we held a stereotypical view of the sysadmin as that guy (and it was always a guy) in the back room of the university computer center who knew everything and could solve all problems by himself. As we ventured into enterprise data centers, we realized the reality was significantly more complex. To describe our findings fully would take a book (which we are currently writing).[6] In this short article, we limit ourselves to a few episodes that illustrate the kinds of collaboration we saw in system administration work and where the major problems lie. As we'll show from real-world stories we collected and our analyses of work patterns, it's really not just one guy in the back room.

The Story of George

George is a Web administrator in a large IT service delivery center. We observed him over a week as he engaged in various planning, deployment, maintenance, and troubleshooting tasks for different customers.[1] George is part of a team of Web administrators; he interacts with

ILLUSTRATION BY YAREK WASZUL

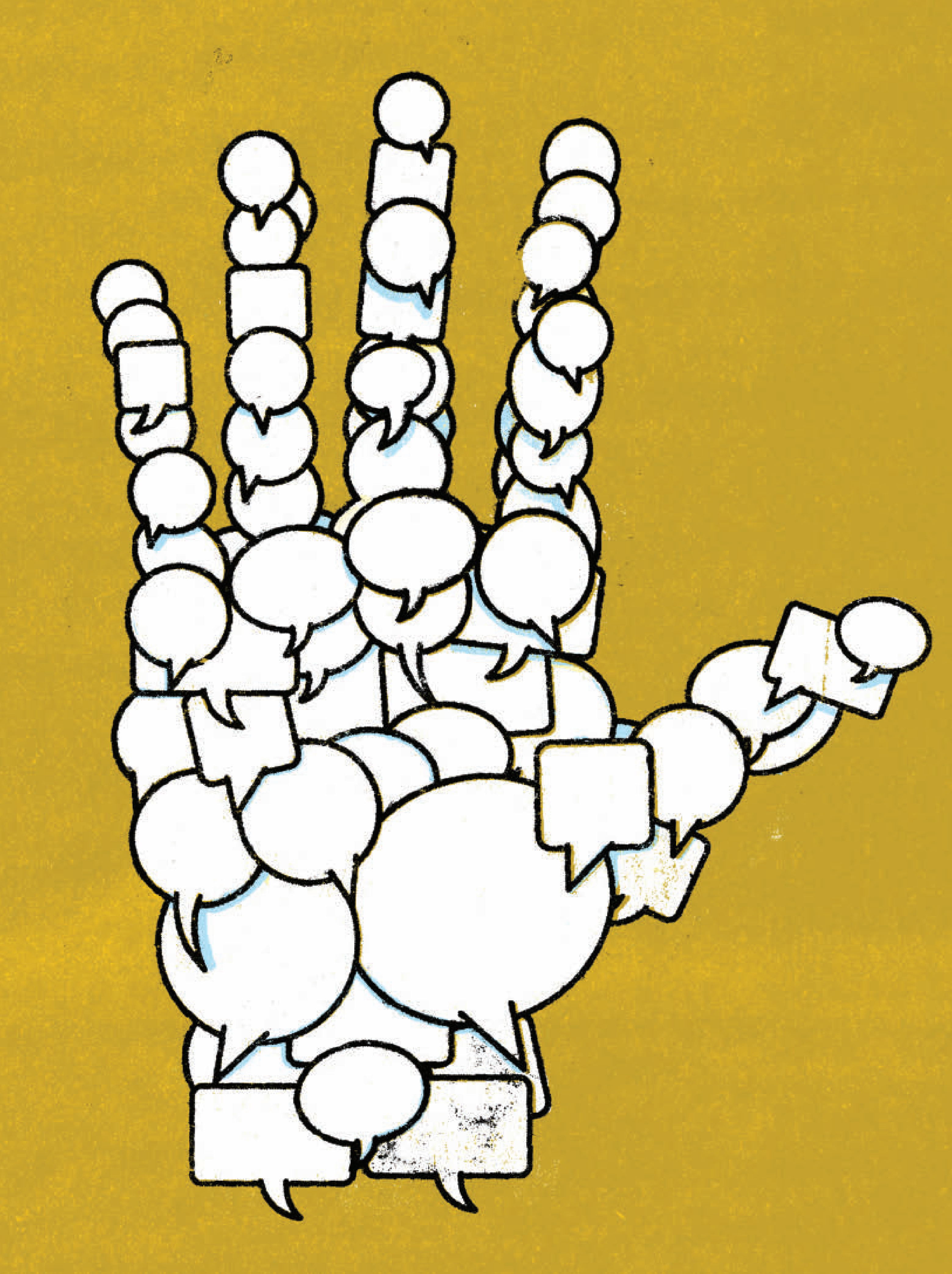
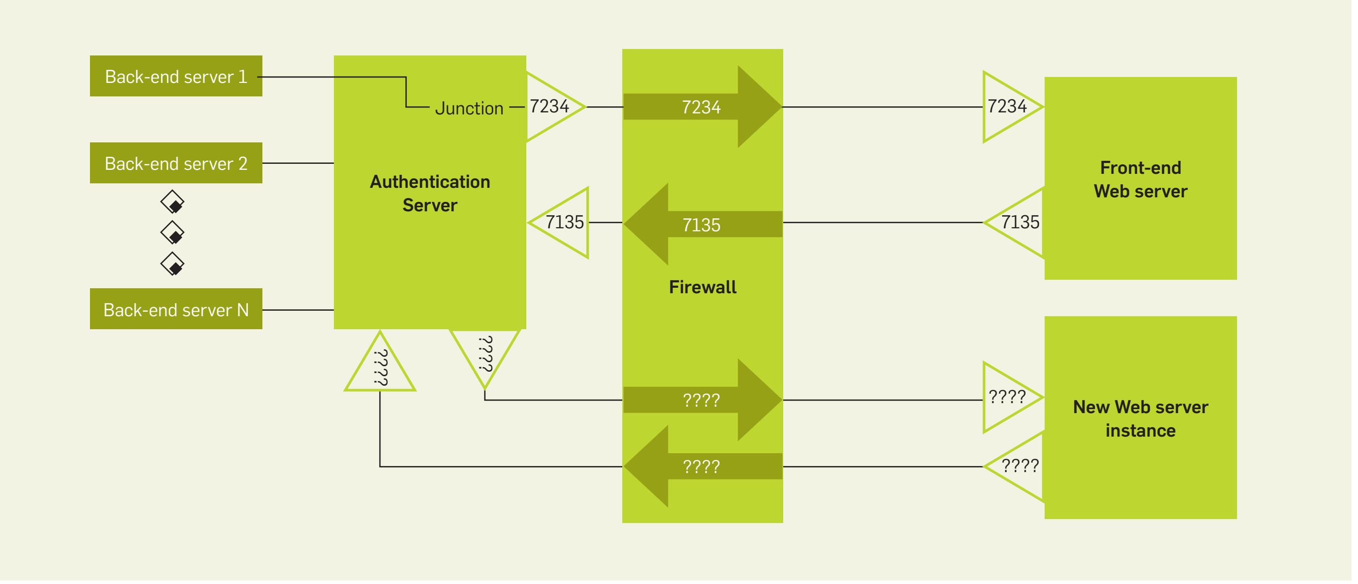


Figure 1. George had to add a new front-end Web server to an existing installation.



the other team members often, as work is distributed. They need to coordinate their actions, hand off long-running tasks, and consult each other (especially during troubleshooting). He also interacts with other teams that are in charge of different areas, such as networks, operating systems, and mail servers.

During our week of observation, one of George's tasks was to set up Web access to email for a customer. This involved creating a new Web-server instance on an existing machine outside the firewall and connecting through a middleware authentication server inside the firewall to a back-end mail server (Figure 1). George had never before installed a second Web server on an existing machine, but he had instructions emailed to him by a colleague as well as access to online documentation. The task involved several people from different teams. Early in the week, George asked the network team to create a new IP address and open ports on the firewall. Throughout the week, we saw him collaborate extensively with Ted, a colleague who was troubleshooting some problems with the authentication server. George's progress was gated by Ted's work, so they exchanged IMs all the time and frequently dropped into each other's offices to work through problems together.

By Friday morning, George had completed all preparations. The final steps should have taken just a few minutes, but this was where the action really began. A mysterious error appeared, and George spent more than two hours troubleshooting the error, mainly in collaboration with others. He had created the new Web-server instance seemingly without incident, and it registered itself with the middleware authentication server. Yet when he issued the command to the middleware server to permit the front-end Web server to talk to the back-end mail server, he got the following message:

```
Error: Could not connect to
server (status: 0x1354a424)
```

Given that three different servers were involved, the error message gave him insufficient information. The online docs and a Web search on the message provided no additional details, so he reached out for help. (For more on error messages, see "Error Messages: What's the Problem?" *ACM Queue*, Nov. 2004.[7])

George's manager suggested calling Adam, the application architect, and George and Adam started troubleshooting together, talking on the phone and exchanging system logs, error messages, configuration files, and sample commands via IMs and email messages (Figure 2). Adam did not have access to the troublesome system, so George acted as his eyes and hands, collecting information and executing commands.

They were not able to find the error, so about an hour in, Adam suggested that George call technical support. He used his office mate's phone (as his own was still connected to Adam), but quickly transitioned communications with tech support to IM. For the next 20 minutes or so, George continued to troubleshoot with Adam on the phone and tech support via IM, and Ted kept popping into the office to offer suggestions. After a while, George became unhappy with the answers from tech support, so Adam hooked him up with one of the developers of the middleware, and they started discussing the problem over IM. Throughout, George remained the sole person with access to the system—all commands and information requests went through him. He became increasingly stressed out as the problem remained unresolved.

Eventually, Ted went back to his own office and looked into the problem independently. He discovered that George had misunderstood one of the front-end server's network configuration parameters, described vaguely in the documentation as "internal port." George thought this parameter (port 7137) specified the port for communication from the front-end to the middleware server, when it went the other way. George, in fact, had made two mistakes: he didn't realize that every front-end server used port 7135 to talk to the middleware server (which was permitted by the firewall, see Figure 1), and he specified a port for communication from the middleware server to the front-end, 7137, that was blocked by the firewall. Communications worked in one direction, but not the other. The software only tested communications in one direction, so the error was not reported until the middleware authentication server was configured. Ted found a solution to this complex situation, and tried unsuccessfully to explain it to George over IM:

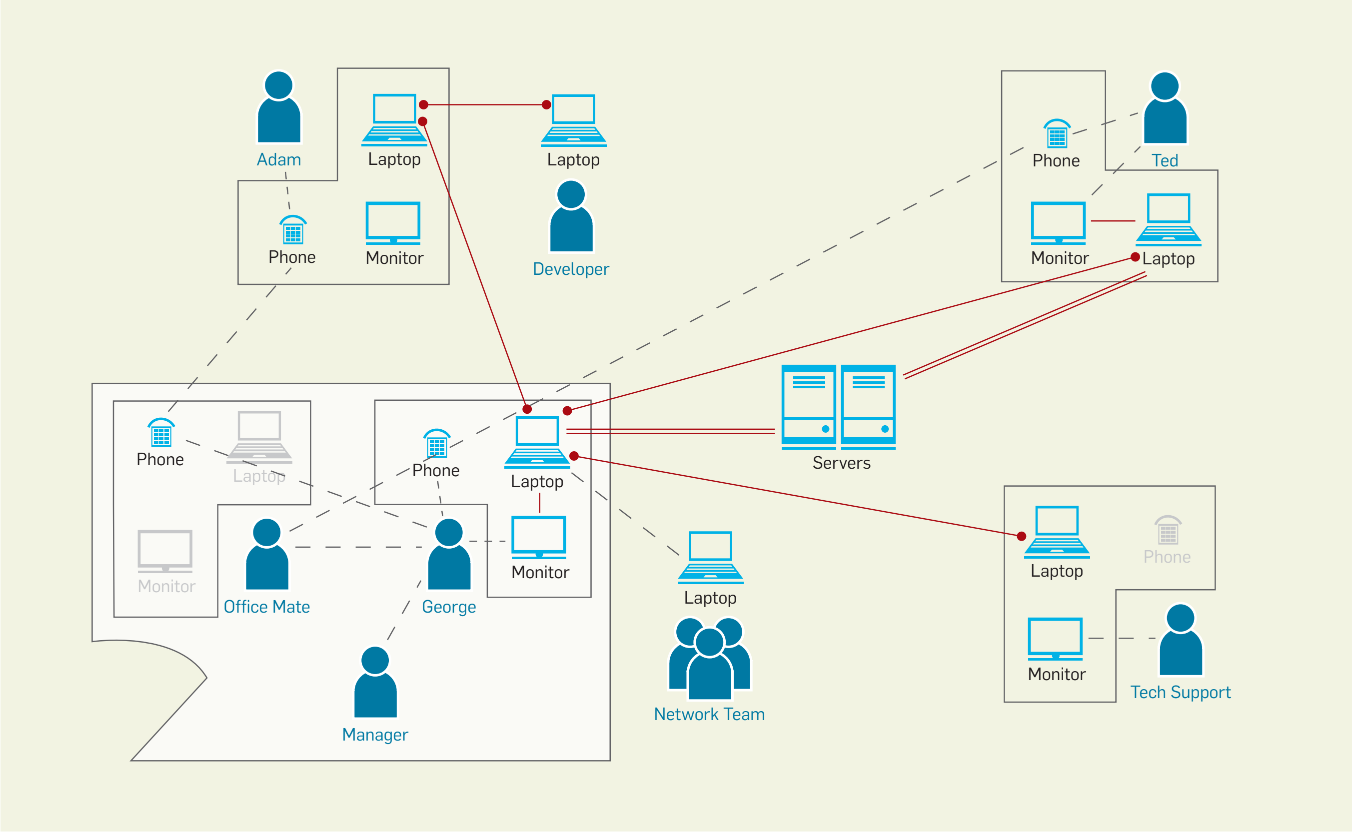
Ted: We were supposed to use 7236. Unconfigure that instance and...
George: Can't specify a return port... you only specify one port.
Ted: You did it wrong.
George: No, I didn't.
Ted: Yes, you did. You need to put in 7236.
George: We just didn't tell it to go both ways. The other port has nothing to do with this.
Ted: Well, all I know is what I see in the conf file.
George: We thought that was the return port. That is not a return port.
Ted: There currently is no listener on [middleware server] on 7137. So use 7236. DO IT!

Ted wasn't getting his point across, and George was getting ever-more frustrated. George told his office mate to call Ted (Adam was still on the other phone), and the conversation immediately switched modes. With the nuance of spoken words, Ted started to realize that George fundamentally misunderstood what was going on. Rather than continually telling George what to do ("DO IT!"), Ted explained why. The task had shifted from debugging the system to debugging George, and they tried to establish a common understanding on which network ports were going which direction.

George: What are you talking about? 7236? We thought that it came in on 7137 and went back on 7236, but we were wrong, that 7236 is like an HTTPS listener port or something?
Ted: It will still come in on 7135 to talk to [middleware] server apparently...
George: Right?
Ted: What's happening is it's actually trying to make a request back, um, through the 72... well, actually trying to make it back through the 7137 to the instance... and it's not happening.
George: I know. I know that. But I can't tell it to...
Ted: Just create it with the 7236. Trust me.
George: Why? That port's not..., that's going the wrong..., that's only one way, too.
Ted: Trust me.
George: It's only one way. Do you understand what I am saying?
Ted: 'Cause it's the [middleware] server talking back to the [Web-server] instance.
George: Yeah, but how does [the Web server] talk to the [middleware] server to make some kind of request?
Ted: 7135 is the standard port it uses in all cases. So we had it wrong. Our assumption on how it works was incorrect.
George: All right, all right.
Ted: If it doesn't work, you can beat me up after.
George: I want to right now. [Laughter on both sides]

How did George get into trouble? Like many failures, there were a number of contributing factors. George misunderstood the meaning of one of the front-end configuration parameters, not realizing that it conflicted with the

Figure 2. George engaged with at least seven different individuals or groups using various means of communication, including instant message (solid lines), email (dashed lines), phone (dotted-and-dashed lines), and face-to-face (dotted lines). Only George and his colleague Ted had direct access to the problematic server (double-solid lines).



firewall rules. The front-end did not test two-way communication, so that errors in the front-end port configuration were not reported until the middleware server was configured. The error message certainly did not help. Perhaps most important was the fact that for most of the troubleshooting session, *George was the only one who had direct access to the system*. All the other participants got their information filtered through George.

Examining the videotapes in detail, we discovered several instances in which George misreported or misunderstood what he saw, filtering the information through his own misunderstanding, and reporting back incorrectly. (One example occurred when George misread the results of a network trace, his misunderstanding filtering out a critical clue.) This prevented Adam and tech support from helping him effectively. The problem was found only when Ted looked at the machine state independently—and then he had to debug George, too. George had many tools for sharing information about system state, but none of them gave the whole picture to the others.

What are the lessons? Collaboration is critical, especially when misunderstandings occur (and from what we saw, incorrect or incomplete understanding of highly complex systems is a common source of problems for sysadmins). Yet collaboration can work only when *correct* information is shared, something that is impeded by misunderstandings and the limitations of communication tools. Proper system design can help avoid misunderstandings in the first place, and improved tools for sharing information could help more quickly rectify misunderstandings when they occur.

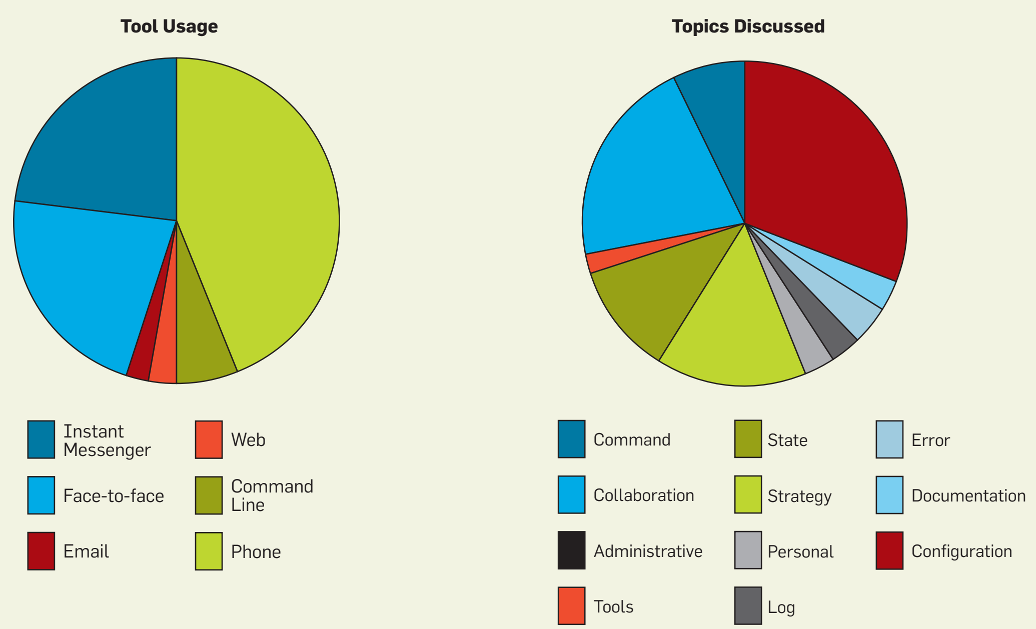
We analyzed the 2.5 hours of George's troubleshooting session, coding each 30-second time slice of what George did (see Figure 3). We found 91% of these time slices were spent in collaboration with other people, either via phone, IMing, email, or face-to-face. Only 6% of the time was he actually interacting with the system, whether to discover state or to make changes, as each interaction was followed by lengthy discussions of the implications of what was seen and what to do next.

While not every troubleshooting episode we witnessed had this extreme level of collaboration, we saw people working together to solve problems much more commonly than a single person toiling alone. We also coded for the topic of collaboration, which included expected topics such as configuration details, system state, ongoing strategy, and what commands to execute. Surprisingly, 21% of the communication involved discussing collaboration itself—for example, "Let me call you" or "Please email me that log file."

Collaboration is especially important in situations where a person's understanding must be debugged, as we saw in George's story. Misunderstandings are a fact of life, and here it was compounded by poorly designed error messages and late reporting of misconfiguration. It can take a long time for someone even to realize that his or her understanding is incorrect. An extra pair of eyes can really help to identify and correct misunderstandings, yet misunderstandings affect what a person reports—so getting a second opinion on the problem will help only if the collaborator gets an accurate picture of the system.

Another lesson is that different communications media are good for different things: the nuance and interchange of the telephone and face-to-face contacts help in getting complex ideas across and in assessing what other people know. IMing is excellent for quickly exchanging commands and error messages verbatim, but subtle personal cues are lost. Even for longtime colleagues like George and Ted, building trust over IM was difficult. Email is great for exchanging lengthy items such as log files and instructions or things that need to persist. Different communications media suggest different levels of commitment to the collaboration.

Figure 3. Accounting of time spent during George's troubleshooting session.



Given the need for collaboration to help sysadmins share their understanding of systems, it is possible to imagine better tools for sharing system state. These tools should take best advantage of different forms of communication to share more completely what is going on with both system and sysadmin alike.

We now turn to another example of collaboration we observed among system administrators working on a much more complex system exhibiting a problem that required incredible effort to understand.

The Crit-Sit

A critical situation, or crit-sit, is a practice that is invoked when an IT system's performance becomes unacceptable and the IT provider must devote specific resources to solving the problem as quickly as possible. Several sysadmins—experts on different components—are brought into a room and told to work together until the problem is fixed. Crit-sits occur more often than sysadmins would like (one we interviewed estimated taking part in four crit-sits per year), and they can last days, weeks, or even months.

We observed one crit-sit for a day, just after it had started, and followed its progress over two months until its solution was found. This was exceptionally long for a crit-sit. It involved an intermittent Web application failure resulting from a subtle interaction of a Web application server and back-end database. Other potential problems were found and fixed along the way, but it took more than 80 days for a dedicated team of experts to determine the true root cause.

At a micro level, being in the room during the crit-sit was fascinating. Eight to 10 people were present in the large conference room, either sitting at the two tables or walking around the room talking; an additional four to six people joined in via conference call and chat room (including technical support representatives for the various software products involved). At first, it seemed amazing to us that this many people had been instructed to work together in a single room until the problem was solved. Indeed, one of the people in the room complained via an instant message to a colleague offsite:

"We're doing lots of PD [problem determination], but nothing that I couldn't have done from home."

Collaboration is especially important in situations where a person's understanding must be debugged, as we saw in George's story. Misunderstandings are a fact of life, and here it was compounded by poorly designed error messages and late reporting of misconfiguration.

After watching the people at work, however, we saw real value in having all of them together in one place. The room was alive with different conversations, usually many at once diverging and rejoining, and with different experts exchanging ideas or asking questions. People would use the whiteboard to diagram theories, and could see and supplement what others were writing. When something important occurred, the attention of everybody in the room was instantly focused. A group chat room was also used as a historical record for system status, error messages, and ideas. Chat was also used for private conversations within the room and beyond, and for exchanging technical information. At one point we saw them build a monitoring script collaboratively through talking, looking at each other's screens, and exchanging code snippets over IM both inside and outside the room.

Not surprisingly, the people in the room appeared much more engaged than the remote participants. Being in the same room signified a level of commitment by the participants. Those on the conference call spoke up only when addressed directly; we assume that they were doing other work and keeping just one ear on the discussions in the room. It is also likely that remote participants could not follow the chaotic, ever-shifting discussions in the room.

At a macro level, following the logs of 11 weeks of troubleshooting was also fascinating. It tells the story of a significant, complicated problem that could not be successfully reproduced on any test system—a problem in which turning on logging would slow the system to the point of unusability at the load levels required to cause the failure. The story shows the crit-sit team interacting with the support teams for a variety of products, escalating to the highest levels, applying patch after patch and experimenting with configuration settings, new hardware, and special versions of the software. The process involved a lot of work by many different teams.

On the whole, the crit-sit was a collaborative effort by a group of experts to understand and repair the behavior of a complex system consisting of many components. They used a wide variety of technical tools: IMing, email, telephone, and screen sharing, yet it seems that they received the greatest value from interacting face-to-face. By being in the same room, people could quickly shift from conversation to conversation when a critical phrase was heard, with a very low barrier to asking someone a question or suggesting an idea.

Although the crit-sit seems heavyweight and wasteful, we have no other approaches that can replicate the collaborative interaction of a bunch of people stuck in a room searching for a solution to a common problem. It would be a revolutionary advance for system administration if a tool were developed that could permit the same engagement in remote collaborators as we saw in the crit-sit room.

We next describe the sorts of collaborations we observed among security administrators at a U.S. university.

One of our motivations for studying sysadmins is the ever-increasing cost of IT management. Part of this can certainly be attributed to the fact that computers get faster and cheaper every year, and people do not.

The "ettercap" Incident

When we first met the security administration team for a computer center at a large university,[5] they seemed somewhat paranoid, making such statements as, "I'll never type my password on a Windows box, because I can't really tell if it's secure." After watching them for two weeks, we realized they had good reason to be cautious. IT systems, as a rule, have no volition and don't care how they're configured or whether you apply a patch to them. Security administrators face human antagonists, however, who have been known to get angry when locked out of a system and work extra hard to find new vulnerabilities and do damage to the data of those who locked them out.

The work of these security administrators was centered around monitoring. New attacks came every week or two. Viruses, worms, and malicious intrusions could happen anytime. They had a battery of automatic monitoring software looking for traces of attacks in system logs and network traffic. Automated intrusion-detection systems needed to err on the side of caution, with the sysadmins making the final decision as to whether suspicious activity was really an attack. These sysadmins relied on communications tools to share information and to help them maintain awareness of what was going on in their center, across their campus, and around the world.

The security administration team shared adjacent offices, so back-and-forth chatter about system activity was common. They joked about taking down the wall to make one big workspace. They also used a universitywide MOO (multiuser domain, object oriented), a textual virtual environment where all the system administrators would hang out, with different "rooms" for different topics. The start of an incident would result in high levels of activity in the security room of the MOO, as security admins from different parts of campus would compare what was happening on their own systems. On a day-to-day basis, the MOO might hold conversations on the latest exploits discovered or theories as to how a virus might be getting into the network. The admins described the MOO's persistence features as really helpful in allowing them to catch up on everything that was going on when they came back after being away, even for a day. They also used a "whisper" feature of the MOO for point-to-point communication (like traditional IM).

An example of MOO use for quick interchange of security status came when we observed a meeting that focused on hacker tools. The security administrators discussed a package called "ettercap." Being unfamiliar with this tool, one of us began searching the Web for information about it using the wireless network. A few minutes later, one of the administrators in the room informed us that a security administrator working remotely had detected this traffic and asked about it on the MOO:

Remote: Any idea who was looking for ettercap? The DHCP logs say [observer's machine name] is a NetBIOS name. Nothing in email logs (like POP from that IP address).
Remote: Seemed more like research.
Remote: The SMTP port is open on that host, but it doesn't respond as SMTP. That could be a hacker defender port.
Local: We were showing how [hacker] downloaded ettercap. One of the visitors started searching for it.
Remote: Ah, OK. Thanks.

In the space of only a few minutes, the sysadmin had detected Web searches for the dangerous ettercap package, identified the name of the machine in question, checked the logs for other activity by that machine, and probed the ports on the machine. He could see that it was probably someone doing research, but checked the MOO to verify that it was in fact legitimate.

The participants also collaborated on a broader scale. During our visit, the site was dealing with a worldwide security incident targeting military, educational, and government sites across the U.S. and Europe. This was a particularly persistent attack—every time an intrusion was detected and a vulnerability was closed, the attackers would come back using a new exploit. The attackers would hop from institution to institution, compromising a machine in one place, collecting passwords, and then trying those passwords on machines at other institutions (as users often have a single password for accounts at different sites).

This broad-based attack required a broad-based response, so security administrators from affected institutions formed an ad hoc community to monitor and share information about the attacks, with the goal of tracing the attacks back to their source. When a compromised machine was found, they would let it remain compromised so that they could then trace the attackers and see where else they were connecting. This collaboration was like information warfare: it was important to share information about known compromised machines and exploits with trusted colleagues, but the information had to be kept from the attackers. You did not want the attackers to know that you had detected their attack and were monitoring their activities. When we first observed them, the security administrators used conference calls for community meetings. Later they found a special encrypted email listserv to keep their information under wraps—but because this tool was unmaintained, they had to adopt and maintain it themselves.

The world of security administration seems very fluid, with new vulnerabilities and exploits discovered every day. Though secrecy was a greater concern than with other sysadmins we observed, collaboration was the foundation of their work: sharing knowledge of unfolding events and system status, especially when an attack might be starting and time was critical.

Conclusion

One of our motivations for studying sysadmins is the ever-increasing cost of IT management. Part of this can certainly be attributed to the fact that computers get faster and cheaper every year, and people do not. Yet complexity is also a huge issue—a Web site today is built upon a dramatically more complicated infrastructure than one 15 years ago. With complexity comes specialization in IT management. With around-the-clock operations needed for today's enterprises, coordination is also a must. System administrators need to share knowledge, coordinate their work, communicate system status, develop a common understanding, find and share expertise, and build trust and develop relationships. System administration is inherently collaborative.

At first, it is easy to think that George's story shows poor debugging practices or worse, poor skills, but we don't think that's the case. The system was complex, the documentation poor, the error messages unenlightening, and no single person was responsible for all of it. Better error messages or better documentation would certainly help, but that misses the point. There will always be cases that go uncovered and complexities that are hidden until it is too late. Modern IT systems are so complex that people will often have an incorrect or incomplete understanding of their operation. That's the nature of IT. The crit-sit story and the security story also show it. The one constant in these cases—and in almost all the cases we observed—was collaboration.

We observed collaboration at many levels: within a small team, within an organization, and across organizations. We observed several different types of collaboration tools in use. We observed people switching from one tool to the other as needs shifted. We also observed simultaneous use of several collaboration tools for different purposes. Not surprisingly, system administrators use the same collaboration tools as the rest of us, but these are not optimized for sysadmin needs—whether it is team brainstorming and debugging or secure information sharing.

Though specific features can be implemented for system administrators, it is clear to us that because of the diverse needs among system administrators, a single collaboration tool will not work for all. There needs to be a variety of tools, and collaboration needs to be a first-class citizen in the work of system administration itself. Better collaboration support could relieve the burden on individuals of communicating and establishing shared context, and so avoiding missed information and enabling a persistent store for communication. We believe that improved tools for system administrator collaboration have great potential to significantly impact system administration work—perhaps even helping to restrain the ever-growing human portion of IT's total cost of ownership. C

Related articles on queue.acm.org

Error Messages: What's the Problem?
Paul P. Maglio, Eser Kandogan
http://queue.acm.org/detail.cfm?id=1036499

Oops! Coping with Human Error in IT Systems
Aaron B. Brown
http://queue.acm.org/detail.cfm?id=1036497

Building Collaboration into IDEs
Li-Te Cheng, Cleidson R.B. de Souza, Susanne Hupfer, John Patterson, Steven Ros
http://queue.acm.org/detail.cfm?id=966803

References

1. Barrett, R., Kandogan, E., Maglio, P.P., Haber, E.M., Prabaker, M., Takayama, L.A. Field studies of computer system administrators: analysis of system management tools and practices. In *Proceedings of the Conference on Computer-Supported Collaborative Work*. 2004.
2. Gartner Group/Dataquest. *Server Storage and RAID Worldwide* (May 1999).
3. Gelb, J.P. System-managed storage. *IBM Systems Journal 28*, 1 (1989), 77–103.
4. ITCentrix. *Storage on Tap: Understanding the Business Value of Storage Service Providers* (Mar. 2001).
5. Kandogan, E., Haber, E. M. 2005. Security and Usability: Designing Secure Systems that People Can Use. In *Security Administration Tools and Practices*. L.F. Cranor and S. Garfinkel, Eds. O'Reilly Media, Sebastapol, 2005, 357–378.
6. Kandogan, E., Maglio, P.P., Haber, E.M., Bailey, J. (forthcoming). *Information Technology Management: Studies in Large-Scale System Administration.* Oxford University Press.
7. Maglio, P.P., Kandogan, E. 2004. Error messages: What's the problem? *ACM Queue 2*, 8 (2004), 50–55.

Eben M. Haber is a research staff member at IBM Research, Almaden, in San Jose, CA. He studies human-computer interaction, working on projects including data mining and visualization, ethnographic studies of IT system administration, and end-user programming tools.

Eser Kandogan is a research staff member at IBM Research, Almaden, San Jose, CA. His interests include human interaction with complex systems, ethnographic studies of system administrators, information visualization, and end-user programming.

Paul P. Maglio is a research scientist and manager at IBM Research, Almaden, San Jose, CA. He is working on a system to compose loosely coupled heterogeneous models and simulations to inform health and health policy decisions. Since joining IBM Research, he has worked on programmable Web intermediaries, attentive user interfaces, multimodal human-computer interaction, human aspects of autonomic computing, and service science.

DOI:10.1145/1866739.1866753

Article development led by acmqueue
queue.acm.org

Talking with Julian Gosper, Jean-Luc Agathos, Richard Rutter, and Terry Coatta.

ACM CASE STUDY

UX Design and Agile: A Natural Fit?

FOUND AT THE intersection of many fields—including usability, human-computer interaction (HCI), and interaction design—user experience (UX) design addresses a software user's entire experience: from logging on to navigating, accessing, modifying, and saving data. Unfortunately, UX design is often overlooked or treated as a "bolt-on," available only to those projects blessed with the extra time and budget to accommodate it. Careful design of the user experience, however, can be crucial to the success of a product. And it's not just window dressing: choices made about the user experience can have a significant impact on a software product's underlying architecture, data structures, and processing algorithms.

To improve our understanding of UX design and how it fits into the software development process, we focus here on a project where UX designers worked closely with software engineers to build *BusinessObjects Polestar* (currently marketed as SAP BusinessObjects Explorer), a business intelligence (BI) query tool designed for casual business users. In the past, such users did not have their own BI query tools. Instead, they would pass their business queries on to analysts and IT people, who would then use sophisticated BI tools to extract the relevant information from a data warehouse. The Polestar team wanted to leverage a lot of the same back-end processing as the company's more sophisticated BI query tools, but the new software required a simpler, more user-friendly interface with less arcane terminology. Therefore, good UX design was essential.

To learn about the development process, we spoke with two key members of the Polestar team: software architect **Jean-Luc Agathos** and senior UX designer **Julian Gosper**. Agathos joined BusinessObjects' Paris office in 1999 and stayed with the company through its acquisition by SAP in 2007. Gosper started working with the company five years ago in its Vancouver, B.C., office. The two began collaborating early in the project, right after the creation of a Java prototype incorporating some of Gosper's initial UX designs. Because the key back-end architecture is one that had been developed earlier by the Paris software engineering team, Gosper joined the team in Paris to collaborate on efforts to implement Polestar on top of that architecture.

To lead our discussion, we enlisted a pair of developers whose skill sets largely mirror those of Agathos and Gosper. **Terry Coatta** is a veteran software engineer who is the CTO of Vitrium Systems in Vancouver, B.C. He also is a member of *ACM Queue*'s editorial advisory board. Joining Coatta to offer another perspective on UX design is **Richard Rutter**, a longtime Web designer and a founder of Clearleft, a UX design firm based in Brighton, England.

Before diving in to see how the collaboration between Agathos and Gosper played out, it's useful to be familiar with a few of the fundamental disciplines of classic UX design:

Contextual inquiry. Before developing use cases, the team observes users of current tools, noting where the pain points lie. Contextual inquiry is often helpful in identifying problems that users are not aware of themselves. Unfortunately, this often proves expensive and so is not always performed.

Formative testing. Formative testing is used to see how well a UX design addresses a product's anticipated use cases. It also helps to determine how closely those use cases actually cleave to real-world experience. In preparation, UX designers generally create lightweight prototypes to represent the use cases the product is expected to eventually service. Often these are paper prototypes, which the test-group participants simply flip through and then comment upon. For the Polestar project, the UX team used a working Java prototype to facilitate early formative testing.

Summative testing. In summative tests, users test-drive the finished software according to some script. The feedback from these tests is often used to inform the next round of development since it usually comes too late in the process to allow for significant changes to be incorporated into the current release.

Although the Polestar team did not have the budget to conduct contextual inquiry, it was able to work closely with the software engineer who built the research prototype responsible for spawning the project. This allowed the team to perform early formative testing with the aid of a working UX design, which in turn made it possible to refine the user stories that would be used as the basis for further testing. Working with the software engineer responsible for the initial design also made it possible to evaluate some of the initial UX designs both from a performance and a feasibility perspective, preventing a lot of unwelcome surprises once development was under way in earnest.

TERRY COATTA: You mentioned you worked early on with a software engineer to iterate on some lightweight paper prototypes. What was the role of the engineer in that process?

JULIAN GOSPER: Adam Binnie, a senior product manager who had conceived of the project, brought me in to elaborate the interaction design of the early Polestar prototype. Davor Cubranic, who had produced that initial proof-of-concept for research purposes, was looking to work some of the user experience ideas I had begun to collaborate on with Adam back into his original design. Davor saw value in creating Java prototypes of some of those new concepts so we would have not only paper prototypes to work with, but also a live prototype that end users could interact with and that development could evaluate from a technical perspective. I really pushed for that since we already had this great development resource available to us. And it didn't seem as though it was going to take all that long for Davor to hammer out some of the key UX concepts, which of course was going to make the live prototype a much better vehicle for formative testing.

Generally speaking, as an interaction designer you don't want to invest a lot of time programming something live, since what you really want is to keep iterating on the fundamentals of the design quickly. That's why working with paper prototypes is so commonplace and effective early in a project. Typically, we'll use Illustrator or Visio to mock up the key use cases and their associated UI, interactions, and task flows, and then output a PowerPoint deck you can just flip through to get a sense for a typical user session. Then various project stakeholders can mark that up to give you feedback. You also then have a tool for formative testing.

Collaborating closely with development at that stage was appealing in this particular case, however, because some of the directions we were taking with the user interface were likely to have serious back-end implications—

Top to Bottom: Jean-Luc Agathos, Terry Coatta, Julian Gosper, and Richard Rutter.

for example, the ability of the application to return and reevaluate facets and visualizations with each click. Having Davor there to help evaluate those proposed design changes right away from a performance perspective through rapid iterations of a lightweight Java-based prototype helped to create a nice set of synergies right from the get-go.

JEAN-LUC AGATHOS: Even though I didn't get involved in the project until later, it seemed as though Davor had produced a solid proof-of-concept. He also figured out some very important processing steps along the way, in addition to assessing the feasibility of some key algorithms we developed a bit later in the process. I think this was critical for the project, since even though people tend to think about user experience as being just about the way things are displayed, it's also important to figure out how that stuff needs to be manipulated so it can be processed efficiently.

GOSPER: That's absolutely correct. For this product to succeed, performance was critical—both in terms of processing a large index of data and being able to evaluate which facets to return to a user to support the experience of clicking through a new data analysis at the speed of thought. The capabilities for assessing the relevance of metadata relative to any particular new query was actually Davor's real focus all along, so he ended up driving that investigation in parallel to the work I was doing to refine the usability of the interface.

I do recall having discussions with Davor where he said, "Well, you know, if you approach 'X' in the way you're suggesting, there is going to be a significant performance hit; but if we approach it this other way, we might be able to get a much better response time." So we went back and forth like that a lot, which I think ultimately made for a much better user experience design than would have been possible had we taken the typical waterfall approach.

COATTA: Are product engineers typically excited about being involved in a project when the user experience stuff is still in its earliest stages of getting sorted out?

AGATHOS: Yes, I think developers need to be acquainted with the user stories as early in the process as possible. To me, user experience and development are essentially one and the same. I see it as our job as a group to turn the user stories into deliverables.

> JULIAN GOSPER
>
> **For this product to succeed, performance was critical—both in terms of processing a large index of data and being able to evaluate which facets to return to a user to support the experience of clicking through a new data analysis at the speed of thought.**

Of course, in development we are generally working from architectural diagrams or some other kind of product description that comes out of product management. We're also looking to the user experience people to figure out what the interaction model is supposed to look like.

The interesting thing about the first version of Polestar is that Julian essentially ended up taking on both roles. He acted as a program manager because he knew what the user stories were and how he wanted each of them to be handled in terms of product functionality. He also had a clear idea of how he wanted all of that to be exposed in the UI and how he wanted end users ultimately to be able to interact with the system. That greatly simplified things from my perspective because I had only one source I had to turn to for direction.

COATTA: You used Agile for this project. At what point in the process can the software developers and UX designers begin to work in parallel?

GOSPER: If you have a good set of user stories that have been agreed upon by the executive members of the project and include clear definitions of the associated workflows and use cases, then the Agile iterative process can begin. At that point you are able to concretely understand the functionality and experience the product needs to offer. On the basis of that, both UX interaction designers and the development team should have enough to get going in parallel. That is, the developers can start working on what the product needs to do while the UX guys can work on use-case diagramming, wireframing and scenarios, as well as begin to coordinate the time of end users to supply whatever validation is required.

The important thing is that you have lots of different people involved to help pull those user stories together. Clearly, the UX team needs to be part of that, but the development team should participate as well—along with the business analysts and anybody else who might have some insights regarding what the product requirements ought to be. That's what I think of when we talk about starting development on the basis of some common "model."

For the most part, development of Polestar's first release went smoothly. Gosper's collaboration with the prototype developer helped iron out some of the technical challenges early on, as well as refine the main user stories, which helped pave the way for implementing the product with Agathos. The user interface that emerged from that process did face certain challenges, which were verified during summative testing, after development work for release 1 was essentially complete. Contributing to the problem was the general dearth of query tools for casual business users at the time. The blessing there was that Gosper and his UX team had the freedom to innovate with their design. That innovation, however, meant the software would inevitably end up incorporating new design concepts that—even with many iterations of formative testing—faced an uncertain reception from users.

Further challenges arose during the development of subsequent releases of Polestar, once Agathos and Gosper were no longer working together. In response to user feedback, the new UX team decided to make fundamental changes to the UI, which forced the engineering team to rearchitect some parts of the software. On top of these challenges, the new team learned what could happen when there is confusion over the underlying conceptual model, which is something Agathos and Gosper had managed to avoid during the first phase of development.

Of course, that's not to say the initial phase was entirely free of development challenges, some of which could be ascribed to pressure to prove the worth of the new endeavor to management.

RICHARD RUTTER: What were the most challenging parts of this project?

AGATHOS: One big challenge was that right after we received the original Java POC (proof of concept), we received word that we needed to produce a Flash version in less than two weeks so it could be shown at Business Objects's [now SAP BusinessObjects] international user group meeting. Actually, that was only one of many constraints we faced. The POC itself imposed certain constraints in terms of how to approach processing the data and exposing appropriate facets to the user. There already were some UI constraints driven by the user stories. *Then* we learned we had only a couple of weeks to get something ready to show to customers. And finally, it was strongly suggested that we use Adobe Flex—something we had not even been aware of previously—to get the work done.

So, our first job was to learn that technology. Initially, we were pretty frustrated about that since it wasn't really as if we had a choice in the matter. Instead of fighting it, however, we quickly realized it was a really good idea. Flex is so powerful that it actually allowed us to come up with a working prototype in time for that user conference.

GOSPER: There was a special aspect to this project in that it was earmarked as innovation—a step toward next-generation self-service BI. Prior to this, the company had never productized a lightweight business intelligence tool for business users, so we didn't really have much in the way of similar products—either within the company or out in the market—that we could refer to. As a result, the design we ended up pushing forward was sort of risky from a user adoption perspective.

The first version of Polestar, in all honesty, tended to throw users for a loop at first. Most of the users we tested needed to spend a few minutes exploring the tool just to get to the point where they really understood the design metaphors and the overall user experience.

AGATHOS: That was definitely the case.

GOSPER: That sparked a fair amount of controversy across the UX group because some of the methodologies around formative testing back then differed from one site to another. The next designers assigned to the project ended up coming to different conclusions about how to refine the interaction design and make it more intuitive. Some questions were raised about whether we were fundamentally taking the right interaction development approach in several different areas of the interface. We employed faceted navigation in a fairly unique way, and the interactions we created around analytics were also pretty novel. [Since users often have only a fuzzy idea about some of the parameters or aspects of the information they're seeking, *faceted navigation* is a UI pattern that facilitates queries by allowing users to interactively refine relevant categories (or "facets") of the search.]

A lot of those initial design directions ended up being re-explored as open topics as the UX team in Paris worked on the second version of the product. They ended up coming to some different conclusions and began to promote some alternatives. That, of course, can be very healthy in the evolution of a product, but at the same time, it can really challenge the development team whenever the new UI choices don't align entirely with the code you've already got running under the hood.

COATTA: Another area where I think there is huge potential for trouble has to do with the feedback process to UX design. I've been through that process and have found it to be extremely challenging. As an example, suppose you need to find two business objects that are related to each other. Let's say we know that one of those objects can be shared across multiple business domains. One possibility is that they share a unique ownership, which would have huge ramifications for the user experience. When we have run across situations like that, we've often had trouble communicating the semantic implications back to the folks who are responsible for doing the UI. I wonder if you've run into similar situations.

AGATHOS: Actually, Julian had already worked all of that out before joining us in Paris, so we didn't have any problems like that during our initial round of development. With later versions, however, we had an issue in the administration module with "infospace," which is a concept we exposed only to administrators. The idea was that you could create an infospace based on some data source, which could be a BWA (Burrows-Wheeler Alignment) index or maybe an Excel spreadsheet. [*BWA* is a fast, lightweight tool that aligns relatively short queries to a sequence database. These sequences are usually indexed in the FASTA format.]

Before anybody can use the system's exploration module to investigate one of these information spaces, that space must be indexed, resulting in a new index entity. That seems straightforward enough, but we spent a lot of time trying to agree on what ought to happen when one person happens to be exploring an infospace while someone else is trying to index the same space. Those discussions proved to be difficult simply because we had not made it explicit that the entity in question was an index. That is, we talked about it at first strictly as an information space. It wasn't until after a few arguments that it became clear what we were actually talking about was an index of that information space.

Whenever the model can be precisely discussed, you can avoid a lot of unnecessary complexity. When a model is correctly and clearly defined right from the outset of a project, the only kind of feedback that ought to be required—and the only sort of change you should need to add to your sprints in subsequent iterations—has to do with the ways you want to expose things. For example, you might decide to change from a dropdown box to a list so users can be given faster access to something—with one click, rather than two. If you start out with a clear model, you're probably not going to need to make any changes later that would likely have a significant impact on either the UI or the underlying architecture.

In developing a product for which there was no obvious equivalent in the marketplace, Agathos and Gosper were already sailing into somewhat uncharted waters. And there was yet another area where they would face the unknown: neither had ever used Agile to implement a UX design. Adding to this uncertainty was the Agile development methodology itself, where the tenets include the need to accept changes in requirements, even late in the development process.

Fortunately, the Polestar team soon embraced the iterative development process and all its inherent uncertainties. In fact, both Gosper and Agathos found Agile to be far more effective than the waterfall methodology for implementing UX designs. This doesn't mean all was smooth sailing, however. Agile requires close collaboration between team members and frequent face-to-face communication, often between stakeholders with vastly different backgrounds and technical vocabularies. It's therefore no surprise that one of the team's biggest challenges had to do with making sure communication was as clear and effective as possible.

> “
> **TERRY COATTA**
> **An area where I think there is huge potential for trouble has to do with the feedback process to UX design. I've been through that process and have found it to be extremely challenging.**
> ”

COATTA: I understand that some UX designers feel Agile is less something to be worked *with* than something to be worked *around*. Yet, this was the first time you faced implementing a UX design using Agile, and it appears you absolutely loved it. Why is that?

GOSPER: In an ideal world, you would do all your contextual inquiry, paper prototyping, and formative testing before starting to actually write lines of code. In reality, because the turnaround time between product inception and product release continues to grow shorter and shorter, that simply isn't possible. If the UX design is to be implemented within a waterfall project, then it's hard to know how to take what you're learning about your use cases and put that knowledge to work once the coding has begun.

In contrast, if you are embedded with the development team and you're acquainted, tactically, with what they're planning to accomplish two or three sprints down the road, you can start to plan how you're going to test different aspects of the user experience in accordance with that.

COATTA: In other words, you just feel your way along?

GOSPER: Yes, and it can be a little scary to dive right into development without knowing all the answers. And by that I don't just mean that you haven't had a chance to work through certain areas to make sure things make sense from an engineering perspective; you also can't be sure about how to go about articulating the design because there hasn't been time to iterate on that enough to get a good read on what's likely to work best for the user. Then you have also got to take into account all the layers of development considerations. All of that comes together at the same time, so you've got to be very alert to everything that's happening around you.

There is also a lot of back-and-forth

in the sprint planning that has to happen. For example, Jean-Luc would let me know we really needed to have a certain aspect of the UI sorted out by some particular sprint, which meant I had essentially received my marching orders. Conversely, there also were times when I needed to see some particular functionality in place in time to use it as part of a live build I wanted to incorporate into an upcoming round of formative testing. That, alone, would require a ton of coordination.

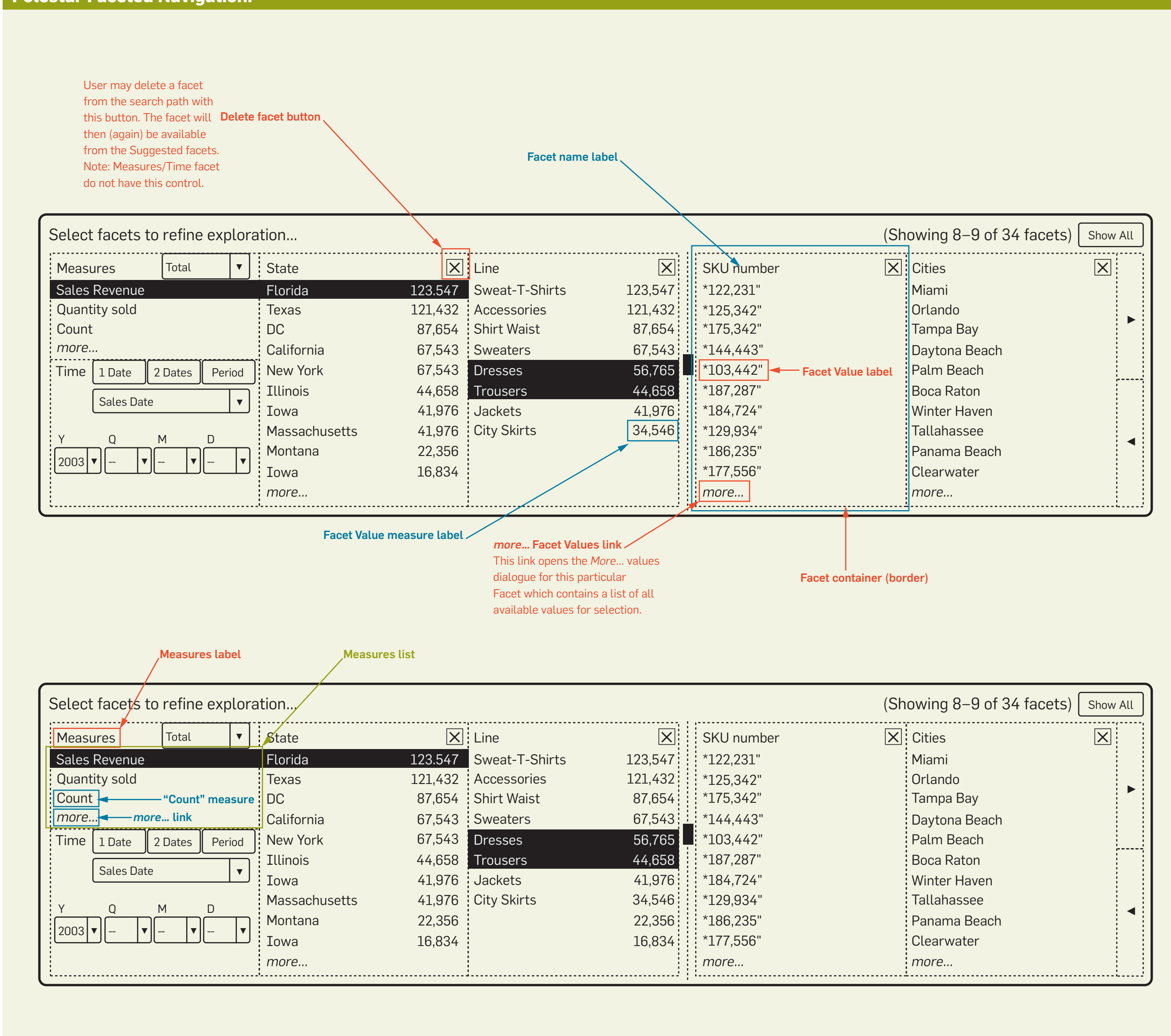
The other important influence on sprint planning comes from product management, since they too often want to be able to show off some certain capabilities by some particular date. With all three of these vested interests constantly vying to have certain parts of the product materialize by certain points in time, there's plenty of negotiating to be done along the way.

RUTTER: Yes, but I think you have to accept that some reengineering of bits and pieces along the way is just an inherent part of the process. That is, based on feedback from testing, you can bank on sprints down the line involving the reengineering of at least a few of the things that have already been built. But as long as that's what you expect...no problem.

AGATHOS: I think you're right about that: we have to develop a mind-set that's accepting of changes along the way. But I actually think the biggest problem is the tooling. The development tools we have today don't help us with all those iterations because they don't provide enough separation between the business logic and the UI. Ideally, we should be able to change the UI from top to bottom and revisit absolutely every last bit of it without ever touching the underlying logic. Unfortunately, today that just isn't the case.

GOSPER: That's why, as a UX interaction designer, you have to be prepared to demonstrate to the product developers that any changes you're recommending are based on substantive evidence, not just some intuitive or anecdotal sense of the users' needs. You need to make a strong case and be able to support design changes with as

Polestar Faceted Navigation.



much quantitative and qualitative end-user validation data as you can get your hands on.

COATTA: So far, we've looked at this largely from the UX side of the equation. What are some of the benefits and challenges of using Agile to implement UX design from more of an engineering perspective?

AGATHOS: The biggest challenge we faced in this particular project—at least after we had completed the first version of Polestar—had to do with changes that were made in the overall architecture just as we were about to finish a release. Even in an Agile environment you still need to think things through pretty thoroughly up front, and then sketch out your design, prototype it, test it, and continue fixing it until it becomes stable. As soon as you've achieved something that's pretty solid for any given layer in the architecture, you need to finish that up and move on to the next higher layer. In that way, it is like creating a building. If you have to go back down to the foundation to start making some fairly significant changes once you've already gotten to a very late stage of the process, you're likely to end up causing damage to everything else you've built on top of that.

The nice thing about Agile is that it allows for design input at every sprint along the way—together with some discussion about the reasoning behind that design input. That's really important. For example, when we were working with Julian, he explained to us that one of the fundamental design goals for Polestar was to minimize the number of clicks the end user would have to perform to accomplish any particular task. He also talked about how that applied to each of the most important user stories. For us as developers, it really helped to understand all that.

I don't think we have exchanges like that nearly enough—and that doesn't apply only to the UX guys. It would also be good to have discussions like that with the program managers.

GOSPER: In those discussions for the Polestar project, one of the greatest challenges for me had to do with figuring out just how much depth to go into when describing a particular design specification. Sometimes a set of wireframes supporting a particular use case seemed to be good enough, since most of what I wanted to communicate could be inferred from them. But there were other times when it would have been helpful for me to break things down into more granular specifications. It was a bit challenging in the moment to sort that out, because on the one hand you're trying to manage your time in terms of producing specifications for the next sprint, but on the other hand you want to get them to the appropriate depth.

The other challenge is that each domain has its own technical language, and it can sometimes prove tricky to make sure you're correctly interpreting what you're hearing. For example, I remember one of the sprints where I was very concerned with a particular set of functionality having to do with users' abilities to specify financial periods for the data they might be looking to explore. I therefore became very active in trying to get product management to allocate more resources to that effort because I was certain it would be a major pain point for end users if the functionality was insufficient.

During that time I remember seeing a sprint review presentation that referred to a number of features, a couple of which related to date and financial period and so forth. Next to each of those items was the notation "d-cut." I didn't say a word, but I was just flabbergasted. I was thinking to myself, "Wow! So they just decided to cut that. I can't believe it. And nobody even bothered to tell me." But of course it turns out "d-cut" stands for "development cut," which means they had already implemented those items. There are times when you can end up talking past each other just because everyone is using terms specific to his or her own technical domain. Of course, the same is true for product and program management as well.

COATTA: Don't the different tools used by each respective domain also make contributions to these communication problems?

AGATHOS: I couldn't agree more. For example, when you program in Java, there are some things you can express and some you cannot. Similarly, for architects, using a language such as UML constrains them in some ways. They end up having to ask themselves tons of questions prior to telling the system what it is they actually want.

COATTA: Since we're talking about how engineers and designers live in different worlds and thus employ different tools, different skill sets, and different worldviews—what do you think of having software engineers get directly involved in formative testing? Does that idea intrigue you? Or does it seem dangerous?

GOSPER: Both, actually. It all depends on how you act upon that information. At one point there is an internal validation process whereby a product that is just about to be released is opened up to a much wider cross section of people in the company than the original group of stakeholders. And then all those folks are given a script that allows them to walk through the product so they can experience it firsthand.

What that can trigger, naturally, is a wave of feedback on a project that is just about finalized, when we don't have a lot of time to do anything about it. In a lot of that feedback, people don't just point out problems; they also offer solutions, such as, "That checkmark can't be green. Make it gray." To take all those sorts of comments at face value, of course, would be dangerous. Anyway, my tendency is to think of feedback that comes through development or any other internal channel as something that should provide a good basis for the next user study.

RUTTER: UX design should always involve contact with lots of different end users at plenty of different points throughout the process. Still, as Julian says, it's what you end up doing with all that information that really matters. When it comes to figuring out how to solve the problems that come to light that way, that's actually what the UX guys get paid to do. ■

Related articles on queue.acm.org

The Future of Human-Computer Interaction
John Canny
http://queue.acm.org/detail.cfm?id=1147530

Human-KV Interaction
Kode Vicious
http://queue.acm.org/detail.cfm?id=1122682

Other People's Data
Stephen Petschulat
http://queue.acm.org/detail.cfm?id=1655240

DOI:10.1145/1866739.1866754

Article development led by acmqueue
queue.acm.org

Managing virtualization at a large scale is fraught with hidden challenges.

BY EVANGELOS KOTSOVINOS

Virtualization: Blessing or Curse?

VIRTUALIZATION IS OFTEN touted as the solution to many challenging problems, from resource underutilization to data-center optimization and carbon emission reduction. However, the hidden costs of virtualization, largely stemming from the complex and difficult system administration challenges it poses, are often overlooked. Reaping the fruits of virtualization requires the enterprise to navigate scalability limitations, revamp traditional operational practices, manage performance, and achieve unprecedented cross-silo collaboration. Virtualization is not a curse: it can bring material benefits, but only to the prepared.

Al Goodman once said, "The perfect computer has been invented. You just feed in your problems and they never come out again." This is how virtualization has come to be perceived in recent years: as a panacea for a host of IT problems. Bringing virtualization into the enterprise is often about *reducing costs* without compromising quality of service. Running the same workloads as virtual machines (VMs) on fewer servers can improve server utilization and, perhaps more importantly, allow the deferral of data-center build-outs—the same data-center space can now last longer.

Virtualization is also meant to enhance the *manageability* of the enterprise infrastructure. As virtual servers and desktops can be live-migrated with no downtime, coordinating hardware upgrades with users or negotiating work windows is no longer necessary—upgrades can happen at any time with no user impact. In addition, high availability and dynamic load-balancing solutions provided by virtualization product families can monitor and optimize the virtualized environment with little manual involvement. Supporting the same capabilities in a nonvirtualized world would require a large amount of operational effort.

Furthermore, enterprises use virtualization to provide IaaS (Infrastructure as a Service) cloud offerings that give users access to computing resources on demand in the form of VMs. This can improve developer productivity and *reduce time to market*, which is key in today's fast-moving business environment. Since rolling out an application sooner can provide first-mover advantage, virtualization can help boost the business.

The Practice

Although virtualization is a 50-year-old technology,[3] it reached broad popularity only as it became available for the x86 platform from 2001 onward—and most large enterprises have been using the technology for fewer than five years.[1,4] As such, it is a relatively new technology, which, unsurprisingly, carries a number of less-well-understood system administration challenges.

Old Assumptions. It is not, strictly speaking, virtualization's fault, but many systems in an enterprise infrastructure are built on the assumption of running on real, physical hardware. The design of operating systems is often based on the principle that the hard disk is local, and therefore reading from and writing to it is fast and low cost. Thus, they use the disk generously in a number of ways, such as caching, buffering, and logging. This, of course, is perfectly fair in a nonvirtualized world.

With virtualization added to the mix, many such assumptions are turned on their heads. VMs often use shared storage, instead of local disks, to take advantage of high availability and load-balancing solutions—a VM with its data on the local disk is a lot more difficult to migrate, and doomed if the local disk fails. With virtualization, each read and write operation travels to shared storage over the network or Fiber Channel, adding load to the network interface controllers (NICs), switches, and shared storage systems. In addition, as a result of consolidation, the network and storage infrastructure has to cope with a potentially much higher number of systems, compounding this effect. It will take years for the entire ecosystem to adapt fully to virtualization.

System Sprawl. Conventional wisdom has it that the operational workload of managing a virtualized server running multiple VMs is similar to that of managing a physical, nonvirtualized server. Therefore, as dozens of VMs can run on one virtualized server, consolidation can reduce operational workload. Not so: the workload of managing a physical, nonvirtualized server is comparable to that of managing a VM, not the underlying virtualized server. The fruits of common, standardized management—such as centrally held configuration and image-based provisioning—have already been reaped by enterprises, as this is how they manage their physical environments. Therefore, managing 20 VMs that share a virtualized server requires the same amount of work as managing 20 physical servers. Add to that the overhead of managing the hypervisor and associated services, and it is easy to see that operational workload will be higher.

More importantly, there is evidence that virtualization leads to an increase in the number of systems—now running in VMs—instead of simply consolidating existing workloads.[2,5] Making it easy to get access to computing capacity in the form of a VM, as IaaS clouds do, has the side effect of leading to a proliferation of barely used VMs, since developers forget to return the VMs they do not use to the pool after the end of a project. As the number of VMs increases, so does the load placed on administrators and on shared infrastructure such as storage, Dynamic Host Configuration Protocol (DHCP), and boot servers.

Most enterprise users of virtualization implement their own VM reclamation systems. Some solutions are straightforward and borderline simplistic: if nobody has logged on for more than three months, then notify and subsequently reclaim if nobody objects. Some solutions are elaborate and carry the distinctive odor of overengineering: analyze resource utilization over a period of time based on heuristics; determine level of usage; and act accordingly. Surprising as it may be there is a lack of generic and broadly applicable VM reclamation solutions to address sprawl challenges. In addition, services that are common to all VMs sharing a host—such as virus scanning, firewalls, and backups—should become part of the virtualization layer itself. This has already started happening with such services entering the hypervisor, and it has the potential to reduce operational workload substantially.

Scale. Enterprises have spent years improving and streamlining their management tools and processes to handle scale. They have invested in a backbone of configuration management and provisioning systems, operational tools, and monitoring solutions that can handle building and managing tens or even hundreds of thousands of systems. Thanks to this—largely home-

grown—tooling, massively parallel operational tasks, such as the build-out of thousands of servers, daily operating system checkouts, and planned datacenter power-downs, are routine and straightforward for operational teams.

Enter virtualization: most vendor solutions are not built for the large enterprise when it comes to scale, particularly with respect to their management frameworks. Their scale limitations are orders of magnitude below those of enterprise systems, often because of fundamental design flaws—such as overreliance on central components or data sources. In addition, they often do not scale out; running more instances of the vendor solution will not fully address the scaling issue, as the instances will not talk to each other. This challenge is not unique to virtualization. An enterprise faces similar issues when it introduces a new operating system to its environment. Scaling difficulties, however, are particularly important when it comes to virtualization for two reasons: first, virtualization increases the number of systems that must be managed, as discussed in the section on system sprawl; second, one of the main benefits of virtualization is central management of the infrastructure, which cannot be achieved without a suitably scalable management framework.

As a result, enterprises are left with a choice: either they live with a multitude of frameworks with which to manage the infrastructure, which increases operational complexity; or they must engineer their own solutions that work around those limitations—for example, the now open source Aquilon framework extending the Quattor toolkit (http://www.quattor.org). Another option is for enterprises to wait until the vendor ecosystem catches up with enterprise-scale requirements before they virtualize. The right answer depends on a number of factors, including the enterprise's size, business requirements, existing backbone of systems and tools, size of virtualized and virtualizable infrastructure, engineering capabilities, and sophistication and size of operational teams.

Interoperability. Many enterprises have achieved a good level of integration between their backbone systems. The addition of a server in the configuration-management system allows it to get an IP address and host name. The tool that executes a power-down draws its data about what to power off seamlessly from the configuration-management system. A change in a server's configuration will automatically change the checkout logic applied to it. This uniformity and tight integration massively simplifies operational and administrative work.

Virtualization often seems like an awkward guest in this tightly integrated enterprise environment. Each virtualization platform comes with its own APIs, ways of configuring, describing, and provisioning VMs, as well as its own management tooling. The vendor ecosystem is gradually catching up, providing increased integration between backbone services and virtualization management. Solutions are lacking, however, that fulfill all three of the following conditions:

- They can be relatively easily integrated with homegrown systems.
- They can handle multiple virtualization platforms.
- They can manage virtual as well as physical infrastructure.

To be sure, some enterprises are fortunate enough to have a homogeneous environment, managed by a product suite for which solid virtualization extensions already exist. In a heterogeneous infrastructure, however, with more than one virtualization platform, with virtualized and nonvirtualized parts, and with a multitude of tightly integrated homegrown systems, the introduction of virtualization leads to *administration islands*—parts of the infrastructure that are managed differently from everything else. This breaks the integration and uniformity of the enterprise environment, and increases operational complexity.

Many enterprises will feel like they have been here before—for example, when they engineered their systems to be able to provision and manage multiple operating systems using the same frameworks. Once again, customers face the "build versus suffer" choice. Should they live with the added operational complexity of administration islands until standardization and convergence emerge in the marketplace, or should they invest in substantial engineering and integration work to ensure hypervisor agnosticism and integration with the existing backbone?

Troubleshooting. Contrary to conventional wisdom, virtualized environments do not really consolidate three physical machines into one physical machine; they consolidate three physical machines onto several physical subsystems, including the shared server, the storage system, and the network.

Finding the cause of slowness in a physical computer is often a case of glancing at a few log files on the local disk and potentially investigating local hardware issues. The amount of data that needs to be looked at is relatively small, contained, and easily found. Monitoring performance and diagnosing a problem of a virtual desktop, on the other hand, requires trawling through logs and data from a number of sources including the desktop operating system, the hypervisor, the storage system, and the network.

In addition, this large volume of disparate data must be *aggregated or linked*; the administrator should be able to obtain information easily from all relevant systems for a given time period, or to trace the progress of a specific packet through the storage and network stack. Because of this multisource and multilayer obfuscation, resolution will be significantly slower if administrators have to look at several screens and manually identify bits of data and log files that are related, in terms of either time or causality. New paradigms are needed for storing, retrieving, and linking logs and performance data from multiple sources. Experience from fields such as Web search can be vital in this endeavor.

Silos? What Silos? In a nonvirtualized enterprise environment, responsibilities for running different parts of the infrastructure are neatly divided among operational teams, such as Unix, Windows, network, and storage operations. Each team has a clear scope of responsibility, communication among teams is limited, and apportioning credit, responsibility, and accountability for infrastructure issues is straightforward.

Virtualization bulldozes these silo walls. Operational issues that involve more than one operational team—and, in some cases, all—become far more common than issues that can be resolved entirely within a silo. As such, cross-silo collaboration and communication are of paramount importance, requiring a true mentality shift in the way enterprise infrastructure organizations operate—as well as, potentially, organizational changes to adapt to this requirement.

Virtualization holds promise as a solution for many challenging problems. Expectations are running high. Can virtualization deliver?

Impact of Changes. Enterprises have spent a long time and invested substantial resources into understanding the impact of changes to different parts of the infrastructure. Change-management processes and policies are well oiled and time tested, ensuring that every change to the environment is assessed and its impact documented.

Once again, virtualization brings fundamental change. Sharing the infrastructure comes with centralization and, therefore, with potential bottlenecks that are not as well understood. Rolling out a new service pack that increases disk utilization by 5IOPS (input/output operations per second) on each host will have very little impact in a nonvirtualized environment—each host will be using its disk a little more often. In a virtualized environment, an increase of disk usage by 5IOPS per VM will result in an increase of 10,000IOPS on a storage system shared by 2,000 VMs, with potentially devastating consequences. It will also place increased load on the shared host, as more packets will have to travel through the hypervisor, as well as the network infrastructure. We have seen antivirus updates and operating-system patches resulting in increases in CPU utilization on the order of 40% across the virtualized plant—changes that would have a negligible effect when applied to physical systems.

Similarly, large-scale reboots can impact shared infrastructure components in ways that are radically different from the nonvirtualized past. Testing and change management processes need to change to account for effects that may be much broader than before.

Contention. Virtualization platforms do a decent job of isolating VMs on a shared physical host and managing resources on that host (such as CPU and memory). In a complex enterprise environment, however, this is only part of the picture. A large number of VMs will be sharing a network switch, and an even larger number of VMs will be sharing a storage system. Contention on those parts of the virtualized stack can have as much impact as contention on a shared host, or more. Consider the case where a rogue VM overloads shared storage: hundreds or thousands of VMs will be slowed down.

Functionality that allows isolating and managing contention when it comes to networking and storage elements is only now reaching maturity and entering the mainstream virtualization scene. Designing a virtualization technology stack that can take advantage of such features requires engineering work and a good amount of networking and storage expertise on behalf of the enterprise customer. Some do that, combining exotic network adapters that provide the right cocktail of I/O virtualization in hardware with custom rack, storage, and network designs. Some opt for the riskier but easier route of doing nothing special, hoping that system administrators will cope with any contention issues as they arise.

GUIs. Graphical user interfaces work well when managing an email inbox, data folder, or even the desktop of a personal computer. In general, it is well understood in the human-computer interaction research community that GUIs work well for handling a relatively small number of elements. If that number gets large, GUIs can overload the user, which often results in poor decision making.[7] Agents and automation have been proposed as solutions to reduce information overload.[6]

Virtualization solutions tend to come with GUI-based management frameworks. That works well for managing 100 VMs, but it breaks down in an enterprise with 100,000 VMs. What is really needed is more intelligence and automation; if the storage of a virtualized server is disconnected, automatically reconnecting it is a lot more effective than displaying a little yellow triangle with an exclamation mark in a GUI that contains thousands of elements. What is also needed is interoperability with enterprise backbones and other systems, as mentioned previously.

In addition, administrators who are accustomed to the piecemeal systems management of the previrtualization era—managing a server here and a storage element there—will discover they will have to adapt. Virtualization brings unprecedented integration and hard dependencies among components—a storage outage could mean that thousands of users cannot use their desktops. Enterprises need to ensure that their operational teams across all silos are comfortable with managing a massively interconnected large-scale system, rather than a collection of individual and independent components, without GUIs.

Conclusion

Virtualization holds promise as a solution for many challenging problems. It can help reduce infrastructure costs, delay data-center build-outs, improve our ability to respond to fast-moving business needs, allow a massive-scale infrastructure to be managed in a more flexible and automated way, and even help reduce carbon emissions. Expectations are running high.

Can virtualization deliver? It absolutely can, but not out of the box. For virtualization to deliver on its promise, both vendors and enterprises need to adapt in a number of ways. Vendors must place strategic emphasis on enterprise requirements for scale, ensuring that their products can gracefully handle managing hundreds of thousands or even millions of VMs. Public cloud service providers do this very successfully. Standardization, automation, and integration are key; eye-pleasing GUIs are less important. Solutions that help manage resource contention end to end, rather than only on the shared hosts themselves, will significantly simplify the adoption of virtualization. In addition, the industry's ecosystem needs to consider the fundamental redesign of components that perform suboptimally with virtualization, and it must provide better ways to collect, aggregate, and interpret logs and performance data from disparate sources.

Enterprises that decide to virtualize strategically and at a large scale need to be prepared for the substantial engineering investment that will be required to achieve the desired levels of scalability, interoperability, and operational uniformity. The alternative is increased operational complexity and cost. In addition, enterprises that are serious about virtualization need a way to break the old dividing lines, foster cross-silo collaboration, and instill an end-to-end mentality in their staff. Controls to prevent VM sprawl are key, and new processes and policies for change management are needed, as virtualization multiplies the effect of changes that would previously be of minimal impact.

Virtualization can bring significant benefits to the enterprise, but it can also bite the hand that feeds it. It is no curse, but, like luck, it favors the prepared.

Acknowledgments

Many thanks to Mostafa Afifi, Neil Allen, Rob Dunn, Chris Edmonds, Robbie Eichberger, Anthony Golia, Allison Gorman Nachtigal, and Martin Vazquez for their invaluable feedback and suggestions. I am also grateful to John Stanik and the *ACM Queue* Editorial Board for their feedback and guidance in completing this article.

Related articles on queue.acm.org

Beyond Server Consolidation
Werner Vogels
http://queue.acm.org/detail.cfm?id=1348590

CTO Roundtable: Virtualization
http://queue.acm.org/detail.cfm?id=1508219

The Cost of Virtualization
Ulrich Drepper
http://queue.acm.org/detail.cfm?id=1348591

References

1. Bailey, M., Eastwood, M., Gillen, A., Gupta, D. Server virtualization market forecast and analysis, 2005–2010. IDC, 2006.
2. Brodkin, J. Virtual server sprawl kills cost savings, experts warn. *NetworkWorld*. Dec. 5, 2008.
3. Goldberg, R.P. Survey of virtual machine research. *IEEE Computer Magazine 7*, 6 (1974), 34–45.
4. Humphreys, J. Worldwide virtual machine software 2005 vendor shares. IDC, 2005.
5. IDC. Virtualization market accelerates out of the recession as users adopt "Virtualize First" mentality; 2010.
6. Maes, P. Agents that reduce work and information overload. *Commun. ACM 37*, 7 (1994), 30–40.
7. Schwartz, B. *The Paradox of Choice*. HarperCollins, NY, 2005.

Evangelos Kotsovinos is a vice president at Morgan Stanley, where he leads virtualization and cloud-computing engineering. His areas of interest include massive-scale provisioning, predictive monitoring, scalable storage for virtualization, and operational tooling for efficiently managing a global cloud. He also serves as the chief strategy officer at Virtual Trip, an ecosystem of dynamic start-up companies, and is on the Board of Directors of NewCred Ltd. Previously, Kotsovinos was a senior research scientist at T-Labs, where he helped develop a cloud-computing R&D project into a VC-funded Internet start-up. A pioneer in the field of cloud computing, he led the XenoServers project, which produced one of the first cloud-computing blueprints.

DOI:10.1145/1866739.1866756

How companies pay programmers when they move the related IP rights to offshore taxhavens.

BY GIO WIEDERHOLD

Follow the Intellectual Property

IN THE ONGOING discussion about offshoring in the computer and data-processing industries, the 2006 ACM report *Globalization and Offshoring of Software* addressed job shifts due to globalization in the software industry.[1] But jobs represent only half of the labor and capital equation in business. In today's high-tech industries, intellectual property (IP) supplies the other half, the capital complement. Offshoring IP always accompanies offshoring jobs and, while less visible, may be a major driver of job transfer. The underlying economic model—involving ownership of profits, taxation, and compensation of workers from the revenue their products generate—has not been explicated and is largely unknown in the computer science community. This article presents the issue of software income allocation and the role IP plays in offshoring. It also tries to explain why computer experts' lack of insight into the economics of software, from investments made, to profits accumulated, to capital becoming available for investment in new projects and jobs.

My intent is to help make computer scientists aware of the relationship of the flow of jobs in computing and the flow of preexisting IP. The ability to create valuable software greatly depends on prior technological prowess. The processes allowing IP to be moved offshore, beyond where the software was created, are formally legal. The resulting accumulation of massive capital in taxhavens[a] has drawn governmental attention and put pressure officials to change tax regulations.[1] However, the changes proposed in these discussions ignore IP's crucial role in generating such capital and, even if enacted, would be ineffective. Transparency is needed to gain public support for any effective change. In addition to advocating transparency about IP transfer, I also offer a radical suggestion—eliminate corporate taxation as a way to avoid the distortion now driving the outflow of IP and providing much of the motivation for keeping capital and IP offshore.

I do not address the risk of misappropriation of IP when offshoring, a related but orthogonal issue, covering instead only the processes that are legal. The risk of loss was addressed throughout the 2006 ACM report,[1] which also cited tax incentives, a much larger economic factor for businesses than misappropriation of IP. The role

a The notion of a taxhaven is a concept in ordinary discourse and a crucial aspect of this article. Moreover, using a one-word term simplifies the specification and parsing of subsets, as in "primary taxhavens" and "semi-taxhavens."

» key insights

- **Profits from the work of software creators and programmers are based on IP being moved offshore.**
- **Locating IP in primary taxhavens damages both developed and emerging economies and disadvantages small businesses.**
- **The capabilities of multinational corporations exceed the capabilities of governments.**

of taxhavens was ignored.

Programmers and the computer scientists supporting their work have traditionally focused on producing quality high-performance software on time and at an affordable cost.[4] They are rarely concerned with the sales and pricing of software, questioning financial policies only when the company employing them goes broke. There is actually a strong sense in the profession that software should be a free good.[12] Implicit in this view is that government, universities, and foundations should pay for software development, rather than the users benefitting from it. In this model, programmers see themselves as artists creating beauty and benefits for all mankind. But consider the size of the software industry. In the U.S. alone, its revenue is $121 billion per year, well over 1% of U.S. GDP.[7] An even larger amount is spent in non-software companies for business-specific software development and maintenance. The more than 4.8 million people employed in this and directly related fields earn nearly $333 billion annually.[5] It is hence unlikely that universal free software is an achievable or even desirable goal. Appropriately, open-source initiatives focus on software that deserves wide public use (such as editors, compilers, and operating systems) and should be freely available to students and innovators.

Flow of Money

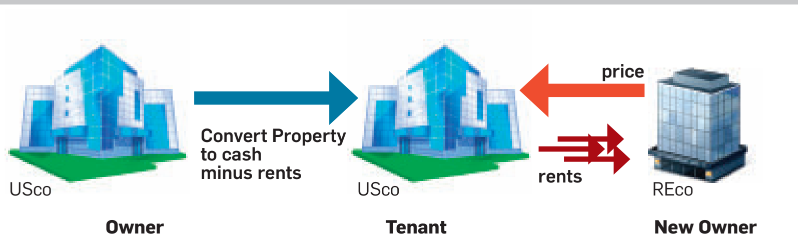
Since economics is the focus of this article, and the economic model of open-source software is not well understood, it is limited to the flow of money related to commercial software, or software written to make a profit, either by selling it or by making enterprises more efficient. Part of the income generated by commercial software is used to pay programmers' salaries. Other portions go to grow the business, to investors, and to taxes that are due and that support the needed infrastructure. Figure 1 sketches the two loops—on top, the flow without IP, and, on bottom, the

ILLUSTRATION BY PETER GRUNDY

Property Rights

A company called *USco* may sell its headquarters building to a real-estate enterprise *REco*, with a provision that *REco* will lease the building back to *USco* (see the figure here). If *USco* receives a fair value for the building, *USco*'s total tangibles remain unchanged until it starts spending the money it received for the sale. *REco* may offer an attractive lease because it is located in a taxhaven. An additional strategy by *USco* is to set up *REco* so it remains under the control of *USco*, also its tenant. Nobody moves, and few employees would notice any change. There may be a new brass plaque on the building and a sign saying "REco" on the door to the rooms housing the people who maintain the building. *USco*'s consolidated annual report, delivered to its shareholders and the IRS, needs to list only the name and location of the controlled subcorporation *REco*; the assets of both are combined. Since the lease receipts and payments cancel out, the more complex financial flow is externally invisible.

Rights to IP sold to an offshore subcorporation.



gains due to having and using IP.

Definitions. Since the revenue aspect of software economics has been ignored in CS curricula, this article introduces several concepts from the literature of business economics and IP generation and exploitation.[25] Within the context of software, many general definitions can be simplified; for instance, manufacturing costs can be ignored, since software products are easy to copy. The cost of equipment is minor when developing and producing software. For tangible products (such as computers), material costs are significant, but for software the cost of tangible media is negligible. The value of software is hence assignable solely to the intellectual effort of its designers, implementers, and marketers. Even the content of a tangible master file and the content in the memory in a cellphone is an intangible. Everything inside the dashed lines of Figure 1 is intangible; only the money surrounding it is real.

If an owner of software protects that ownership, the software is considered IP. To protect its IP an enterprise would disallow purchasers of copies to make further copies that might be sold. The means of protection vary, including asserting copyrights, registering trademarks, making copying difficult, only releasing binary images of the code, and threatening prosecution. The IP held within the owning enterprise is primarily protected by keeping the source code secret.

Employees and contractors in the software industry are routinely required to sign nondisclosure agreements in order to protect trade secrets. Trade secrets cover the majority of the IP owned by companies developing software. Patents can protect visible processes (such as one-click-ordering). But patenting internal processes that contribute to the creation of quality software would require revealing the methods, records, and documents employed. Such things are best protected as trade secrets.[10] For companies that market software, trademarks represent a complementary component of corporate IP. Trademarks are visible and registered and their use defended. The value of trademarks derives from a combination of having excellent products in the market, marketing methods to grow sales, and advertising to spread the word. For products that benefit from ongoing sales, customer lists are also of value and protected as a trade secret. Employees are motivated to keep trade secrets by the contribution to their collective job security provided by these constraints.

Without protected IP, a company's income would be at the routine level provided by commodity products, with margins after production and distribution of, say, 10%, insufficient to invest in innovation. Having IP without a knowledgeable staff to exploit it is equally futile.[17] When a high-tech company is acquired, senior staff are typically required to remain until its processes are solidly embedded within the purchaser. Even a startup, with no identifiable IP, will have some specific ideas and concepts in the minds of its founders that form the seeds of growth. Time and money are required for such seeds to mature into IP and then into salable products, a delay referred to as "economic gestation time" or "lag."[30]

Protected IP and staff knowledge and expertise within a company are the intangibles that together represent the enterprise's IP. The employees who know how to exploit it complement the IP; such integration is essential for success. The combination of labor and IP leads to non-routine profits. The margins are then typically greater than 50%, even after spending on R&D, allowing further investment and growth.

IP and Jobs

All subsequent developers on a software project benefit from the work that has gone before, that is, from the IP already in place. That IP complements the knowledge due to education and prior experience new employees bring to the job.

The importance of IP to employee productivity becomes clear when companies grow to a size that offshore-outsourcing of jobs is considered. The new offshore workers, whether testers providing quality assurance, maintenance programmers, sales staff, or call-center employees, receive material representing IP that exists in the parent company at the time. Offshore researchers also build on requirements for innovation and the experience collected by the parent company.

Splitting Intellectual Capital

Intellectual capital is the know-how of the work force and the IP it has generated. As a company matures its IP grows and becomes its major asset. Risk of IP loss due to employee turnover becomes less critical. To gain financial flexibility, a company might identify and isolate its IP. The rights to identified IP, as trademarks and technology, can be moved to a distinct subcorporation. Separating IP is an initial phase in setting up an offshored operation when significant IP is involved.[29] To be productive, the extant technology still must be made available to the creative workers, by having the productive corporate divisions pay license fees to the subcorporation holding the technology IP; see the sidebar "Property Rights" for an illustrative example clarifying the process of splitting rights from the property itself.

Such transfer-of-rights transactions are even simpler when applied to IP. The rights to a company's IP or to an arbitrary fraction of that IP can be sold to a controlled foreign holding company (CFH) set up in a taxhaven. Once the rights to the IP are in the CFH the flow of income and expenses changes. The rights to the IP are bundled, so no specific patents, trade secrets, or documents are identified. The net income attributable to the fraction of the IP held in the CFH is collected in an account also held in the taxhaven. One way of collecting such income is to charge royalties or license fees for the use of the IP at the sites where the workers create saleable products, both at home and offshore. There is no risk of IP loss at the CFH, because nothing is actually kept there. To reduce the risk of IP loss where the work is performed, new offshore sites are set up as controlled foreign corporations (CFCs), rather than using contractors.[29] Since IP is crucial to making non-routine profits, the royalty license fees to be paid to the CFH can be substantial and greatly reduce the profitability at the parent and at the CFCs from worldwide software product sales (see Figure 2).

The consolidated enterprise thus gains much strategic business flexibility. Work can be shifted wherever it appears to be effective, perhaps where new incentives are provided, and the needed IP can be made available there, as long as the license fees are paid to the CFH.[22] Paying these fees as royalties on profits is preferred, since profits reflect the ever-changing profit margins due to sales variability and to switching to cheaper labor.

The actual IP content needed to perform creative work is transferred through multiple paths: documents, code, and personal interaction by staff interchanges among the remote sites and the originating location. Most transfers are mediated by the Internet, allowing rapid interaction and feedback. The CFH does not get involved at all.

Three types of parties are involved in IP creation (as in Figure 2): the parent, the CFH, and the CFCs. Employees work at and create IP at the parent and the CFC locations. Large multinational corporations actually establish dozens of controlled entities to take advantage of different regulations and incentives in various countries.

Valuing Transferred IP

The CFH subcorporation that obtains the rights to the IP, and that will profit

Figure 1. Components of the economic loops for software.

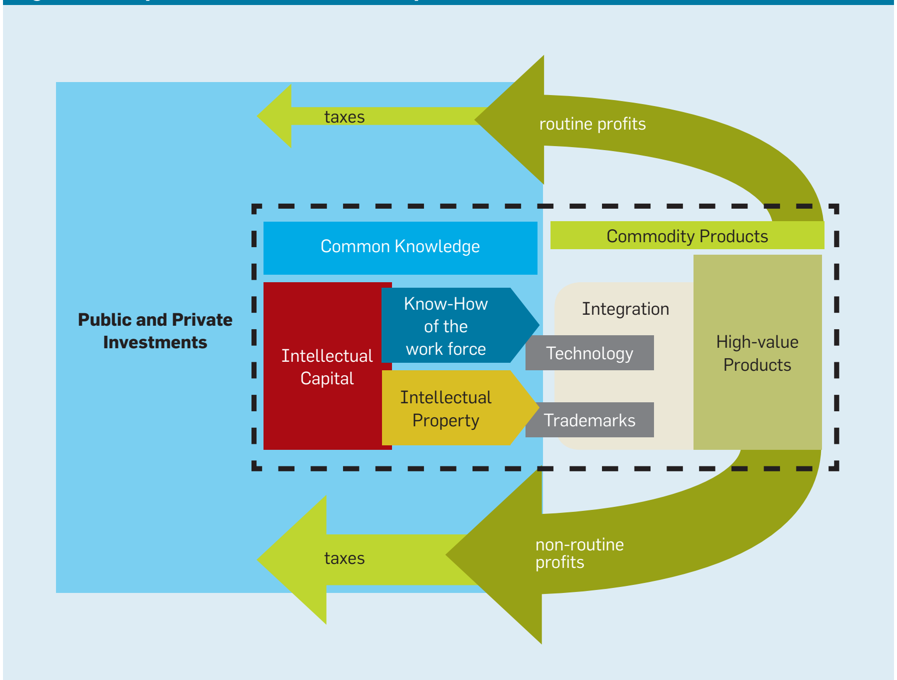
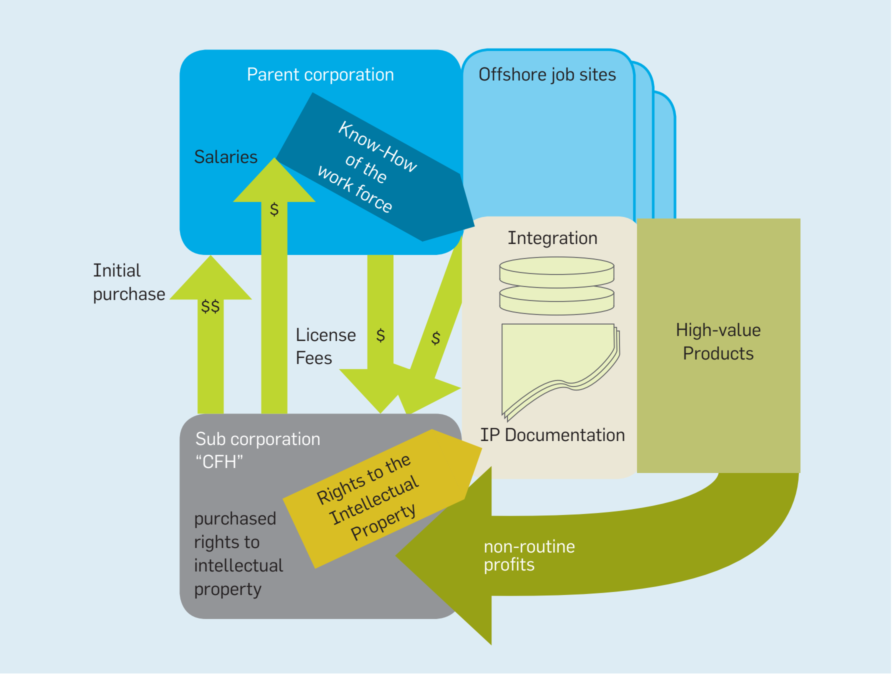


Figure 2. Extracting and selling the rights to derive income from a property.



from fees for its use, must initially purchase the IP from the prior owner. For work that is offshored, the new workers do not contribute prior proprietary knowledge, only IP subsequently. But setting a fair price for the initial IP received is difficult and risky. If it is overvalued the company selling it to the CFH will have gained too much income, on which it must pay taxes. If it is undervalued, excessive profits will accrue to the CFH.

How does the company document the value of its transferred IP? The annual reports to shareholders and the 10-K reports submitted annually to the U.S. Securities and Exchange Commission rarely include estimates of the value of a company's intangible property. Only when one company acquires another high-tech company are due-diligence assessments of the IP obtained made. Various method types help assess the value of the transferred IP from a parent to its CFCs or CFH are available, including these five (with many variations):

Future income. Predict the future income ceded to the CFH, subtract all expected costs, and reduce the remainder to account for routine profits. Compute the IP's net present value (NPV) over its lifetime to obtain the amount due to the IP[2];

Shareholder expectations. Use shareholder expectations embodied in the company's total market capitalization, subtract the value of its tangibles, and split the remainder among the CFH and the parent[3];

Expected value. Search for similar public transactions where the IP transfer was among truly independent organizations, then adjust for differences and calculate a median value[19];

Diminishing maintenance. Aggregate the NPV of the specific incomes expected from the products sold over their lifetimes as their initial IP contribution due to maintenance diminishes[31];

Expected R&D margin. Extrapolate the past margin obtained from ongoing R&D investment as it delivers benefits over successive years.[15]

All valuation methods depend on data. The availability of trustworthy data determines applicability and the trustworthiness of their results. Using more than one method helps increase confidence in the resulting valuation of the IP. If existing trademarks are being transferred or kept after an acquisition their contribution to income requires adjustments as well. The benefits of marketing expenses tend to be short-lived. Technological IP is a mixture, some created through product improvement that drives revenue with little delay and some resulting from the fundamental R&D that takes a long time to get to market.

While valuing all IP in a company is a challenge, for the purpose of offshoring software IP, simplification of confounding items is possible, making the task easier. The amount of tangible property is relatively small in a high-tech company. The value of the work force can be determined through comparison with public data of acquisitions of similar companies with little IP.

Taxhavens

Offshoring is greatly motivated by being able to avoid or reduce taxes on income by moving rights to the IP into low-tax jurisdictions, or taxhavens, categorized as semi-taxhavens, or countries looking to attract jobs through active external investments, and primary taxhavens. Semi-taxhavens tend to provide temporary tax benefits. Countries intent on growth like Israel and Ireland have offered tax holidays to enterprises setting up activities there, while India provides incentives for companies that export. Many Eastern European countries have set up or are considering similar initiatives. Setting up a subsidiary CFC in a semi-taxhaven requires financial capital and significant corporate IP, helping workers be productive quickly. These resources are best provided via primary taxhavens.

Primary taxhavens are countries with small populations that focus on attracting companies that will not use actual resources there, and with no local personnel hired. Although their role is crucial in offshoring, the jobs issue is not raised, and the services needed for remote holding companies (such as registration with local government, mail forwarding, and arranging boards-of-directors meetings) are offered by branches of global accounting firms. For example, a single well-known five-floor building in the Cayman Islands is the address for 18,000 holding companies, and the entire country, with fewer than 50,000 inhabitants, hosts more than 90,000 registered companies and banks. The income from the $3,000 annual registration fees for that many companies allows the Cayman Islands to not impose any taxes on anybody. Even the beach resorts, available for board of directors meetings, are not taxed.

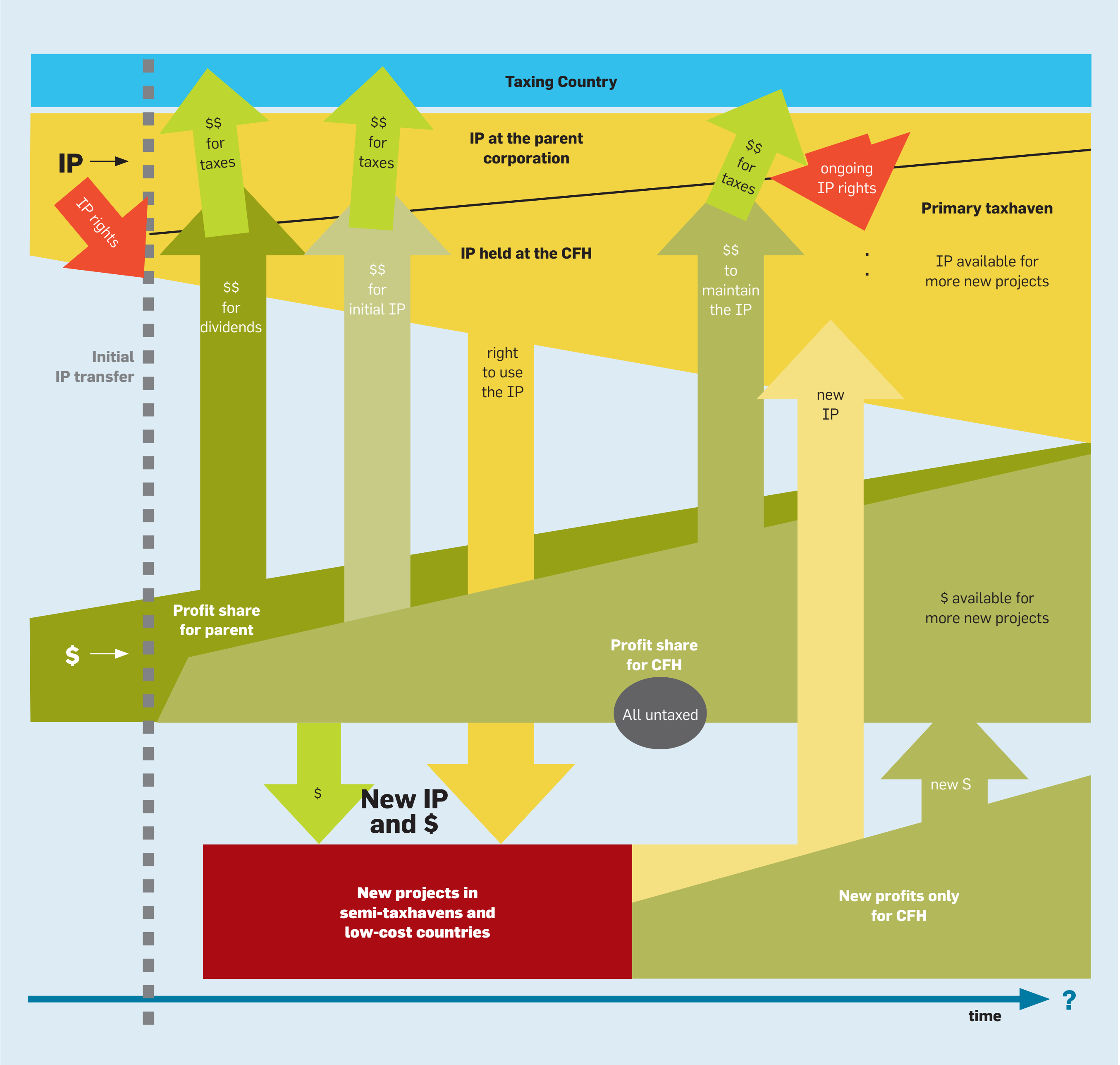
Defining what makes a country a prime taxhaven varies but always includes negligible or no taxation and lack of transparency. A few dozen jurisdictions actively solicit and lobby for business, citing their taxhaven advantages. Reporting income and assets is often not required. Advantages can be combined; for instance, the rule that Cayman-based corporations must have one local annual meeting can be overcome by having a Cayman company be formally resident in a British Crown Colony like Bermuda. Often, only a single CFH shareholder is fully controlled by another corporation. Cayman companies need not have external directors on their boards, and optional board meetings can be held anywhere convenient . Neither audits nor annual reports are required, but for criminal cases, records are made available. At the extreme end of the taxhaven spectrum are countries identified by the Organisation for Economic Co-operation and Development as uncooperative taxhavens, even sheltering fraud.[20]

The use of primary taxhavens causes a loss to the U.S. Treasury of more than $100 billion annually, a substantial amount compared to the $370 billion total actually collected as corporate tax.[8,32] Only $16 billion in taxes were paid (for the year reviewed) by multinational corporations in the U.S.; smaller businesses pay the greatest share.[27] The actual tax rate paid by companies using taxhavens averages 5%, even as they complain about high U.S. corporate taxes. It was estimated at a G-20 meeting that developing nations overall lose annual revenue of $125 billion due to taxhaven use.[11]

Assets in a Taxhaven

Following an IP transfer to a primary taxhaven, the taxhaven CFH will have two types of assets: more-auditable fi-

Figure 3. The changing investment scene, as taxhaven resources become available.



nancial ones, derived from licensing and royalties for use of the IP, and the IP itself. Both grow steadily, as outlined in Figure 3 and are now freely available to initiate and grow projects in any CFC. The IP in the primary taxhaven is made available by charging license fees to projects in the semi-taxhavens, providing immediate income to the CFH. When the projects have generated products for sale, royalties on the sales provide further income to the CFH.

Initially, the income at the CFH is used to reimburse the parent company for the assumed value of the IP that was transferred,[19] an amount typically paid over several years. Moving the IP offshore early in the life of a company, when there is little documented IP, increases the leverage of this approach. The income of the CFH is also used to pay for ongoing R&D or for the programmers at the parent company and in any IP-generating offshore location.[28] U.S. taxes must be paid on such funds as they are repatriated to the U.S., since they represent taxable income. The funds not needed to support R&D (often more than half after the initial payback) can remain in the CFH. In each yearly cycle yet more funds flow to the holding company in the taxhaven. Additional funds may be repatriated from a CFH when a country (such as the U.S.) offers tax amnesties for capital repatriation or when the parent companies show losses, so the corporate income tax due can be offset.[1,6]

The payments by the CFH for creative work ensure that all resulting IP belongs to the CFH. While the value of the initial IP purchased diminishes over time, the total IP held in the CFH increases as the product is improved and provides a long-term IP and income stream.

Starting an Offshore Project

Money and IP accumulated in a primary taxhaven should be deployed for generating yet more income and to avoid showing excess capital on the consolidated books. Money will be needed to pay for workers on new projects, and IP is needed to make them effective and bring future income to non-routine levels. The period covered by Figure 3 may extend 10 years or more. The value of the IP needed for a new project is based on the expectation of the income it will generate and is very high for a promising project. The export of IP, just like any property export, should generate income to the provider. Such exported income, moved via a primary taxhaven, avoids any payment of taxes. Note that only an appropriate fraction of the rights to the IP is shipped out of the taxhaven. The actual documents are provided by the originators, wherever they might work, but the documents are kept by the CFH in the taxhaven, which formally owns the IP.

Since the value of IP is not reported anywhere, nothing is visible to employees or shareholders, except a few financial experts in the company, or more typically, their financial advisors.

If new projects are initiated fully by the primary taxhaven and not located with the parent, both the IP exports and resulting non-routine income enabled by IP transfers escape all taxation. In most jurisdictions no regulatory authority will check if the IP valuations and related royalties are fair.[21] Funding a similar new project at a taxing locale that requires visibility, as in the U.S., would be costly and awkward; for instance, profit margins would be out of line and raise suspicion. Investing in low-cost countries that tax profits still provides tax benefits, since high license fees paid to a CFH greatly reduce the taxable profit in those countries.

Over time, the share of profits directly available to the parent decreases, and dividends may have to be paid out of CFH funds. These payments are taxed twice, first as part of corporate taxes and then, at a greatly reduced rate, as shareholder income. Paying few dividends out of CFH funds and starting new projects instead is an attractive alternative.

Eliminate corporate taxation as a way to avoid the distortion now driving the outflow of IP and providing much of the motivation for keeping capital and IP offshore.

Financial and IP Assets

Several issues should be of concern to computer professionals, even though their effects are indirect. For example, three effects of moving IP rights to a taxhaven are instability of work opportunities, imbalance of large-versus-small companies, and loss of infrastructure support.

Having funds in a primary taxhaven gives multinational corporations enough flexibility to exploit global opportunities. Whenever and wherever business opportunities and incentives are available, funds can be deployed quickly. Of course, moving work to semi-taxhavens is more advantageous than supporting work in countries that tax at typical rates. The flexibility for high-tech industries is notably great compared to industries that rely on tangible assets. When semi-taxhaven countries attract investment in tangibles (such as a car factory), benefits are retained after the tax holiday, but IP investments can be redeployed quickly. Only a few senior people may have to move (physically). When the semi-taxhaven also has a low-wage structure, benefits for the consolidated corporation multiply.

Countries seeking jobs for growing populations are pleased about any investment that creates jobs, even if structured to minimize local corporate profits and taxes. Governments often create semi-taxhavens to encourage new projects but rarely realize how rapidly corporations can move facilities that depend primarily on IP. Temporary tax incentives then fail to provide the long-term benefits these countries expected in return for the tax losses.

The tax avoidance enabled by accumulating IP and funds in any taxhaven reduces the ability of governments of the countries where the actual work is performed to support the infrastructure they need for a healthy economy. That infrastructure includes public roads and transportation, health services, and education for future generations. Scarcities can be seen in Silicon Valley, California, Silicon Glen, Scotland, and Electronics City, Bangalore,[1] but tracing cause and effect is complex.

Smaller companies that have not had the opportunity to employ taxhavens are disadvantaged, even though most economists view them as the major drivers of growth. In addition to unequal taxation, they are also less likely to be able to benefit from government tax credits for R&D. Such credits enable mature corporations to offset their U.S. R&D labor costs against any taxes remaining on their profits, while the IP generated accrues to their CFH. Smaller companies establish tax shelters as well. Since most large companies have already established them, tax-consulting firms intent on their own growth now also market taxhavens to midsize businesses.

Lack of Transparency

The creators of the software, even if they care where their paycheck comes from and where the IP they produce goes, cannot follow the tortuous path from sales to salary.[18] Many intermediate corporate entities are involved, so tracing the sources of programmer income becomes well nigh impossible. Even corporate directors, despite having ultimate responsibility, are not aware of specifics, other than having agreed to a tax-reduction scheme operated by their accountants. Investors and shareholders will not find in consolidated annual reports or 10-K filings any direct evidence of taxhaven use, since regulations devised to reduce paperwork hide amounts held and internal transactions within controlled corporations. Funds transferred for R&D and dividends from taxhavens are first deposited in corporate income accounts, then taxed, but may remain eligible for government tax credits for corporate research. The taxpayers in these countries are not aware of benefits beyond salaries; that is, income from profitable IP will not accrue to the country providing the research credits.[23]

Tax-avoidance processes have been explored in many publications but not applied to corporate IP transfer[16]; the adventures of movie and sports stars make more interesting reading. Promoters of corporate tax reduction, seeking to, perhaps, gain more business, provide general documentation, and even address the risks of misvaluation of IP and of faulty royalty rates.[19] The complexity of this arrangement makes it easy to cross the boundaries of legality. Misvaluations can greatly reduce the magnitude of IP exports and consequent tax benefits. The firms that provide advice for setting up tax shelters have the required broad competencies not available in the critical constituencies of the computing community.[1] Staff of firms providing such advice often function invisibly as directors of their customers' CFHs. Most advising organizations protect themselves from legal liability by formally splitting themselves into distinct companies for each country in which they operate. These companies then rejoin by becoming members of a "club" (*Vereinsgesetz*) set up under Swiss laws. The member companies of such clubs do not assume responsibility for one another's work and advice. But the club can share resources, information, and income among member companies, allowing them to function as a unit.

U.S. government officials are restricted in how they share corporate information. Rules established to protect corporate privacy prohibit the sharing of information among Internal Revenue staff regarding arrangements used by specific taxpayers to avoid taxes. Even a 2008 U.S. government report[14] had to rely on survey data and could not use corporate filings. A thorough study of IP and capital flow would require changes in the restricting regulations.

Incremental Suggestions

No matter what conclusions you draw from this article, any follow-up will require increased transparency. U.S. Senate bill S.506 introduced in March 2009 by Senator Carl Levin (Dem., Michigan) "To restrict the use of taxhavens..." includes measures to increase access to corporate data of companies that set up taxhavens and to the information their advisers provide. Its primary goal is to tax CFHs as if they were domestic corporations. It is unclear if the bill will become law, since confounding arguments can be raised about its effects. The role of IP and jobs is not addressed in the bill, and unless the public is well-informed, meaningful reforms will have difficulty gaining traction.

Without transparency one cannot quantitatively assess the relationships between IP offshoring and jobs offshoring. While it is clear that there is initial dependency, long-term effects are only imagined. Tax schemes clearly create an imbalance of actual tax-rates being paid by small- versus large-business innovators.

With more information in hand, scientists and researchers in industry might try to influence potentially worrisome corporate policies. While employees have few (if any) legal rights to determine corporate directions, they may well have expectations about their employers' behavior. A corporation may listen, since the motivation of its work force is a valuable asset. Corporate leaders might not have considered the long-term effect of schemes they themselves set in place to minimize taxes. However, these leaders are also under pressure to compete, nationally and internationally.[1] It has been suggested that international initiatives are needed to level the corporate-taxation playing field.

Change the Flow

A radical solution to problems created by tax-avoidance schemes is to do away with corporate taxation altogether and compensate the U.S. government for the loss of tax income by fully taxing dividends and capital gains, that is, by imposing taxes only when corporate profits flow to the individuals consuming the benefits. The net effect on total tax revenues in the U.S. might be modest, since, in light of current tax avoidance strategies, corporations contributed as little as 8% to total U.S. tax revenue in 2004.[6] Such a radical change would reduce the motivation for many distortions now seen in corporate behavior. Small businesses unable to pay the fees and manage the complexity of taxhavens would no longer be disadvantaged.

Getting effective international agreement seems futile, and no single government can adequately regulate multinational enterprises. Unilateral alternatives to deal with countries that shelter tax-shy corporations are infeasible as well, even without consideration of the role of IP and malfeasance.[9] An underlying problem is that the law equates a corporation with a person, allowing confusing arguments, even though people have morals, motivations, and obligations that differ greatly from the obligations of corporations. This equivalence is seen as a philosophical mistake by some.[13] For instance, humans cannot, without creating corporate entities, split themselves into multiple clones that take advantage of differing taxation regimes. In practice, not taxing corporations is such a radical change, affecting so many other aspects of the economy and public perception, that it is as unlikely as many other proposed tax reforms.[8]

Why Care?

The knowledge-based society brought forth a revolution of human productivity in the past 50 years, moving well beyond the industrial revolution that started more than a century earlier, and globalization is a means to distribute its benefits. But the growth of assets in taxhavens deprives workers worldwide of reasonably expected benefits. These hidden assets have grown to be a multiple of annual industry revenue, exceeding the assets held in the countries where the IP is being created. The presence of significant IP rights in taxhavens provides global corporations great flexibility to invest capital anywhere, avoiding income due to IP from being taxed anywhere. The combination of reduced support for education, government research funding, and physical infrastructure, along with the increased motivation to start new initiatives in semi-taxhavens and the imbalance of small businesses versus global corporations, is bound to affect the future of enterprises in countries that initiated high-tech industries, though the rate and final magnitude is unpredictable today. Better-educated scientists will be less affected and feel the effects more slowly.[26] But any industry requires a mix of related competencies. It took 50 years for the U.S. car industry to be reduced to its current state. The velocity of change when intangibles, instead of tangible capabilities, are involved may well be greater.

The large amount of capital accumulated in taxhavens encourages ever-greater investment in foreign companies. As of August 2010, such investment was reported to amount to $7.6 billion, a 168% increase from the same period in 2009.[23] The eight largest companies have $300 billion available in taxhavens. Cisco Systems alone reported it had $30 billion available in its tax shelters and expects to keep spending on foreign acquisitions. Such investment will create jobs all over the world, primarily in semi-taxhavens.

More support for CS education was a major emphasis of the ACM report, but where will the funding come from? The taxes on Cisco's available funds, were they to be used for investment in the U.S., exceed total funding for the National Science Foundation and Defense Advanced Research Projects Agency. The IP, if it remains offshore, would quickly refill Cisco's coffers there. Discussions concerning future education, leading to growth of knowledge-based industries, job creation, protection of retirement benefits, and the required infrastructure for growing businesses are futile if the creators of the required intellectual resources are uninformed about the interaction of IP and capital allocation. Initiating effective action is more difficult still.

Acknowledgments

This exposition was motivated by the Rebooting Computing meeting in Silicon Valley in January 2009 (http://www.rebootingcomputing.org/content/summit) and benefited from discussions on the topic with Peter J. Denning (the organizer), Joaquin Miller, Erich Neuhold, Claudia Newbold, Shaibal Roy, Stephen Smoliar, Shirley Tessler, Andy van Dam, Moshe Y. Vardi, and unknown, patient *Communications* reviewers. Any remaining lack of clarity is due to my failure in presenting a novel topic adequately to a technical audience. Any errors and opinions are also solely my responsibility.

References

1. Aspray, W., Mayadas, F., and Vardi, M.Y., Eds. *Globalization and Offshoring of Software.* A Report of the ACM Job Migration Task Force. ACM, New York, 2006.
2. Babcock, H. *Appraisal Principles and Procedures.* American Society of Appraisers, Herndon, VA, 1994; first edition 1968.
3. Becker, B. Cost sharing buy-ins. Chapter in *Transfer Pricing Handbook, Third Edition*, R. Feinschreiber, Ed. John Wiley & Sons, Inc., New York, 2002, A3–A16.
4. Boehm, B. *Software Engineering Economics.* Prentice-Hall, Upper Saddle River, NJ, 1981.
5. Bureau of Labor Statistics. *National Employment and Wage Data Survey.* Bureau of Labor Statistics, Washington D.C., May 2007; http://www.bls.gov/oes/
6. Clausing, K. *The American Jobs Creation Act of 2004, Creating Jobs for Accountants and Lawyers.* Urban-Brookings Tax Policy Center, Report 311122, Washington, D.C., Dec. 2004.
7. Compustat. *Financial Results of Companies in SIC Code 7372 for 1999 to 2002*; http://www.compustat.com
8. Cray, C. Obama's tax haven reform: Chump change. *CorpWatch* (June 15, 2009); http://www.corpwatch.org/article.php?id=15386.
9. Dagan, T. The tax treaties myth. *NYU Journal of International Law and Politics 32* article 379181 (Oct. 2000), 939–996.
10. Damodaran, A. *Dealing with Intangibles: Valuing Brand Names, Flexibility and Patents Working Paper.* Stern School of Business Reports, New York University, New York, Jan. 2006; http://pages.stern.nyu.edu/~adamodar/
11. Deutsche Stiftung für Entwicklungsländer. Bitter losses. English version (Aug. 7, 2010); http://www.inwent.org/ez/articles/178169/index.en.shtml
12. Gay, J. *Free Software, Free Society: In Selected Essays of Richard M. Stallman.* GNU Press, Boston, MA, 2002.
13. Gore, A. *The Assault on Reason.* Bloomsbury Publications, London, 2008.
14. Government Accountability Office. *International Taxation: Large U.S. Corporations and Federal Contractors with Subsidiaries in Jurisdictions Listed as Tax Havens or Financial Privacy Jurisdictions.* U.S. Government Accountability Office Report GAO-09-157, Washington, D.C., Dec. 2008; http://www.gao.gov/new.items/d09157.pdf
15. Grossman, G.M. and Helpman, E. *Innovation and Growth in the Global Economy, Seventh Edition.* MIT press, Cambridge, MA, 2001.
16. Johnston, D.C. *Perfectly Legal.* Portfolio Publishers, New York, 2003.
17. Kiesewetter-Koebinger, S. Programmers' capital. *IEEE Computer 43*, 2 (Feb. 2010), 106–108.
18. Lev, B. *Intangibles, Management, Measurement and Reporting.* Brookings Institution Press, Washington, D.C., 2001.
19. Levey, M.M., Wrappe, S.C., and Chung, K. *Transfer Pricing Rules and Compliance Handbook.* CCH Wolters Kluwer Publications, Chicago, IL, 2006.
20. Makhlouf, G. *List of Uncooperative Tax Havens.* OECD's Committee on Fiscal Affairs. Organisation for Economic Co-operation and Development, Paris, France, Apr. 19, 2002.
21. Parr, R. *Royalty Rates for Licensing Intellectual Property.* John Wiley & Sons, Inc., New York, 2007.
22. Rahn, R.W. In defense of tax havens. *The Wall Street Journal* (Mar. 17, 2009).
23. Rashkin, M. *Practical Guide to Research and Development Tax Incentives: Federal, State, and Foreign, Second Edition.* CCH Wolters Kluwer Publications, Chicago, IL, 2007.
24. Saitto, S. U.S. tech firms shop abroad to avoid taxes. *Bloomberg Businessweek* (Sept. 6, 2010), 31–32.
25. Smith, G. and Parr, R. *Intellectual Property, Valuation, Exploitation, and Infringement Damages*, John Wiley & Sons, Inc., New York, 2005.
26. Tambe, P.B. and Hitt, L.M. How offshoring affects IT workers. *Commun. ACM 53*, 10 (Oct. 2010), 62–70.
27. Tichon, N. *Tax Shell Game: The Taxpayer Cost of Offshore Corporate Havens.* U.S. Public Interest Research Group, Apr. 2009; http://www.uspirg.org/news-releases/tax-and-budget/tax-and-budget-news/washington-d.c.-taxpayers-footing-a-100-billion-bill-for-tax-dodgers
28. Weissler, R. *Advanced Pricing Agreement Program Training on Cost Sharing Buy-In Payments. Transfer Pricing Report 533.* IRS, Washington D.C., Feb. 2002.
29. Wiederhold, G., Tessler, S., Gupta, A., and Smith, D.B. The valuation of technology-based intellectual property in offshoring decisions. *Commun. Assoc. Infor. Syst. 24*, 31 (June 2009), 523–544.
30. Wiederhold, G. Determining software investment lag. *Journal of Universal Computer Science 14*, 22 (2008).
31. Wiederhold, G. What is your software worth? *Commun. ACM 49*, 9 (Sept. 2006), 65–75.
32. Wilson, S. Is this the end for treasure islands? *MoneyWeek* (Mar. 13, 2009).

Gio Wiederhold (gio@cs.stanford.edu) is Professor (Emeritus) of Computer Science, Medicine, and Electrical Engineering at Stanford University, Stanford, CA; http://infolab.stanford.edu/people/gio.html

DOI:10.1145/1866739.1866757

The ICE abstraction may take CS from serial (single-core) computing to effective parallel (many-core) computing.

BY UZI VISHKIN

Using Simple Abstraction to Reinvent Computing for Parallelism

THE RECENT DRAMATIC shift from single-processor computer systems to many-processor parallel ones requires reinventing much of computer science to build and program the new systems. CS urgently requires convergence to a robust parallel general-purpose platform providing good performance and programming easy enough for all CS students and graduates. Unfortunately, ease-of-programming objectives have eluded parallel-computing research over at least the past four decades. The idea of starting with an established easy-to-apply parallel programming model and building an architecture for it has been treated as radical by hardware and software vendors alike. Here, I advocate an even more radical parallel programming and architecture idea: Start with a simple abstraction encapsulating the desired interface between programmers and system builders.

key insights

- **Computing can be reinvented for parallelism, from parallel algorithms through programming to hardware, preempting the technical barriers inhibiting use of parallel machines.**
- **Moving beyond the serial von Neumann computer (the only successful general-purpose platform to date), computer science will again be able to augment mathematical induction with a simple one-line computing abstraction.**
- **Being able to think algorithmically in parallel is a significant advantage for systems developers and programmers building and programming multi-core machines.**

Parallel Algorithms

The following two examples explore how these algorithms look and the opportunities and benefits they provide to systems developers and programmers.

Example 1. **Given are two variables A and B, each containing some value. The exchange problem involves exchanging their values; for example, if the input to the exchange problem is A=2 and B=5, then the output is A=5 and B=2. The standard algorithm for this problem uses an auxiliary variable X and works in three steps:**

```
X:=A
A:=B
B:=X
```

In order not to overwrite A and lose its content, the content of A is first stored in X, B is then copied to A, and finally the original content of A is copied from X to B. The work in this algorithm is three operations, the depth is three time units, and the space requirement (beyond input and output) is one word.

Given two arrays A[0..n-1] and B[0..n-1], each of size *n*, the array-exchange problem involves exchanging their content, so A(i) exchanges its content with B(i) for every i=0..n-1. The array exchange serial algorithm serially iterates the standard exchange algorithm n times. Here's the pseudo-code:

```
For i =0 to n−1 do
        X:=A( i ) ; A( i ):=B( i ) ; B( i ):=X
```

The work is 3n, depth is 3n, and space is 2 (for X and i). A parallel array-exchange algorithm uses an auxiliary array X[0..n-1] of size n, the parallel algorithm applies concurrently the iterations of the serial algorithm, each exchanging A(i) with B(i) for a different value of i. Note the new pardo command in the following pseudo-code:

```
For i =0 to n−1 pardo
        X( i ):=A( i ) ; A( i ):=B( i ) ; B( i ):=X( i )
```

This parallel algorithm requires 3n work, as in the serial algorithm. Its depth has improved from 3n to 3. If the size of the array n is 1,000 words, it would constitute speedup by a factor of 1,000 relative to the serial algorithm. The increase in space to 2n (for array X and n concurrent values of i) demonstrates a cost of parallelism.

Example 2. **Given is the directed graph with nodes representing all commercial airports in the world. An edge connects node u to node v if there is a nonstop flight from airport u to airport v, and s is one of these airports. The problem is to find the smallest number of nonstop flights from s to any other airport. The WD algorithm works as follows: Suppose the first i steps compute the fewest number of nonstop flights from s to all airports that can be reached from s in at most i flights, while all other airports are marked "unvisited."**

Step i+1 concurrently finds the destination of every outgoing flight from any airport to which the fewest number of flights from s is exactly i, and for every such destination marked "unvisited" requires i+1 flights from s. Note that some "unvisited" nodes may have more than one incoming edge. In such a case the arbitrary CRCW convention implies that one of the attempting writes succeeds. While we don't know which one, we do know all writes would enter the number i+1; in general, however, arbitrary CRCW also allows different values.

The standard serial algorithm for this problem[9] is called breadth-first search, and the parallel algorithm described earlier is basically breadth-first search with one difference: Step i+1 described earlier allows concurrent-writes. In the serial version, breadth-first search also operates by marking all nodes whose shortest path from s requires i+1 edges after all nodes whose shortest path from s requires i edges. The serial version then proceeds to impose a serial order. Each newly visited node is placed in a first-in-first-out queue data structure.

Three lessons are drawn from this example: First, the serial order obstructs the parallelism in breadth-first search; freedom to process in any-order nodes for which the shortest path from s has the same length is lost. Second, programmers trained to incorporate such serial data structures into their programs acquire bad serial habits difficult to uproot; it may be better to preempt the problem by teaching parallel programming and parallel algorithms early. And third, to demonstrate the performance advantage of the parallel algorithm over the serial algorithm, assume that the number of edges in the graph is 600,000 (the number of nonstop flight links), and the smallest number of flights from airport s to any other airport is no more than five. While the serial algorithm requires 600,000 basic steps, the parallel algorithm requires only six. Meanwhile, each of the six steps may require longer wall clock time than each of the 600,000 steps, but the factor 600,000/6 provides leeway for speedups by a proper architecture.

I begin by proposing the Immediate Concurrent Execution (ICE) abstraction, followed by two contributions supporting this abstraction I have led:

XMT. A general-purpose many-core explicit multi-threaded (XMT) computer architecture designed from the ground up to capitalize on the on-chip resources becoming available to support the formidable body of knowledge, known as parallel random-access machine (model), or PRAM, algorithmics, and the latent, though not widespread, familiarity with it; and

Workflow. A programmer's workflow links ICE, PRAM algorithmics, and XMT programming. The ICE abstraction of an algorithm is followed by a description of the algorithm for the synchronous PRAM, allowing ease of reasoning about correctness and complexity, which is followed by multithreaded programming that relaxes this synchrony for the sake of implementation. Directly reasoning about soundness and performance of multithreaded code is generally known to be error-prone. To circumvent the likelihood of errors, the workflow incorporates multiple levels of abstraction; the programmer must establish only that multithreaded program behavior matches the synchronous PRAM-like algorithm it implements, a much simpler task. Current XMT hardware and software prototypes and demonstrated ease-of-programming and strong speedups suggest that CS may be much better prepared for the challenges ahead than many of our colleagues realize.

A notable rudimentary abstraction—that any single instruction available for execution in a serial program executes immediately—made serial computing simple. Abstracting away a hierarchy of memories, each with greater capacity but slower access time than its predecessor, along with different execution time for different operations, this Immediate Serial Execution (ISE) abstraction has been used by programmers for years to conceptualize serial computing and ensure support by hardware and compilers. A program provides the instruction to be executed next at each step (inductively). The left side of Figure 1 outlines serial execution as implied by this ISE abstraction, where unit-time instructions execute one at a time.

The rudimentary parallel abstraction I propose here is that indefinitely many instructions available for concurrent execution execute immediately, dubbing the abstraction Immediate Concurrent Execution. A consequence of ICE is a step-by-step (inductive) explication of the instructions available next for concurrent execution. The number of instructions in each step is independent of the number of processors, which are not even mentioned. The explication falls back on the serial abstraction in the event of one instruction per step. The right side of Figure 1 outlines parallel execution as implied by the ICE abstraction. At each time unit, any number of unit-time instructions that can execute concurrently do so, followed by yet another time unit in which the same execution pattern repeats, and so on, as long as the program is running.

How might parallelism be advantageous for performance? The PRAM answer is that in a serial program the number of time units, or "depth," is the same as the algorithm's total number of operations, or "work," while in the parallel program the number of time units can be much lower. For a parallel program, the objective is that its work does not much exceed that of its serial counterpart for the same problem, and its depth is much lower than its work. (Later in the article, I note the straightforward connection between ICE and the rich PRAM algorithmic theory and that ICE is nothing more than a subset of the work-depth model.) But how would a system designer go about building a computer system that realizes the promise of ease of programming and strong performance?

Outlining a comprehensive solution, I discuss basic tension between the PRAM abstraction and hardware implementation and a workflow that goes through ICE and PRAM-related abstractions for programming the XMT computer architecture.

Some many-core architectures are likely to become mainstream, meaning they must be easy enough to program by every CS major and graduate. I am not aware of other many-core architectures with PRAM-like abstraction. Allowing programmers to view a computer operation as a PRAM would make it easy to program,[10] hence this article should interest all such majors and graduates.

Until 2004, standard (desktop) computers comprised a single processor core. Since 2005 when multi-core computers became the standard, CS has appeared to be on track with a prediction[5] of 100+-core computers by the mid-2010s. Transition from serial (single-core) computing to parallel (many-core) computing mandates the reinvention of the very heart of CS, as these highly parallel computers must be built and programmed differently from the single-core machines that dominated standard computer systems since the inception of the field almost 70 years ago. By 2003, the clock rate of a high-end desktop processor had reached 4GHz, but processor clock rates have improved only barely, if at all, since then; the industry simply did not find a way to continue improving clock rates within an acceptable power budget.[5] Fortunately, silicon technology improvements (such as miniaturization) allow the amount of logic a computer chip can contain to keep growing, doubling every 18 to 24 months per Gordon Moore's 1965 prediction. Computers with an increasing number of cores are now expected but without significant improvement in clock rates. Exploiting the cores in parallel for faster completion of a computing task is today the only way to improve performance of individual tasks from one generation of computers to the next.

Unfortunately, chipmakers are designing multi-core processors most programmers can't handle,[19] a problem of broad interest. Software production has become a key component of the manufacturing sector of the economy. Mainstream machines most programmers can't handle cause significant decline in productivity of manufacturing, a concern for the overall economy. Andy Grove, former Chairman of the Board of Intel Corp., said in the 1990s that the software spiral—the cyclic process of hardware improvements leading to software improvements leading back to hardware improvements—was an engine of sustained growth for IT for decades to come. A stable application-software base that could be reused and enhanced from one hardware generation to the next was available for exploitation. Better performance was assured with each new generation, if only the hardware could run serial code faster. Alas, the software spiral today is broken.[21] No broad parallel-computing application software base exists for which hardware vendors are committed to improving performance. And no agreed-upon parallel architecture allows application programmers to build such a base for the foreseeable future. Instating a new software spiral could indeed be a killer app for general-purpose many-core computing; application software developers would put it to good use for specific applications, and more consumers worldwide would want to buy new machines.

This robust market for many-core-based machines and applications leads to the following case for government support: Foremost among today's challenges is many-core convergence, seeking timely convergence to

Figure 1. Serial execution based on the serial ISE abstraction vs. parallel execution based on the parallel ICE abstraction.

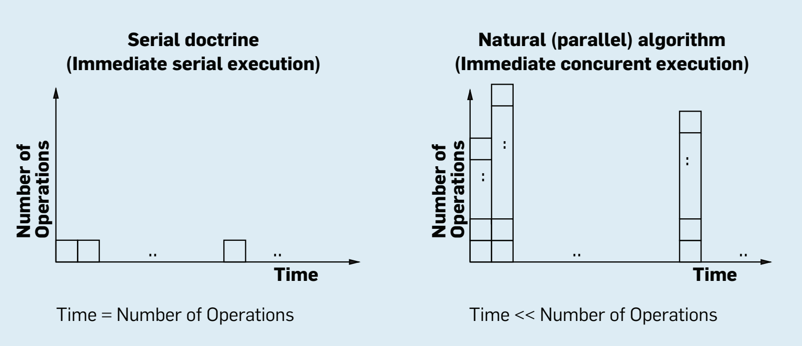
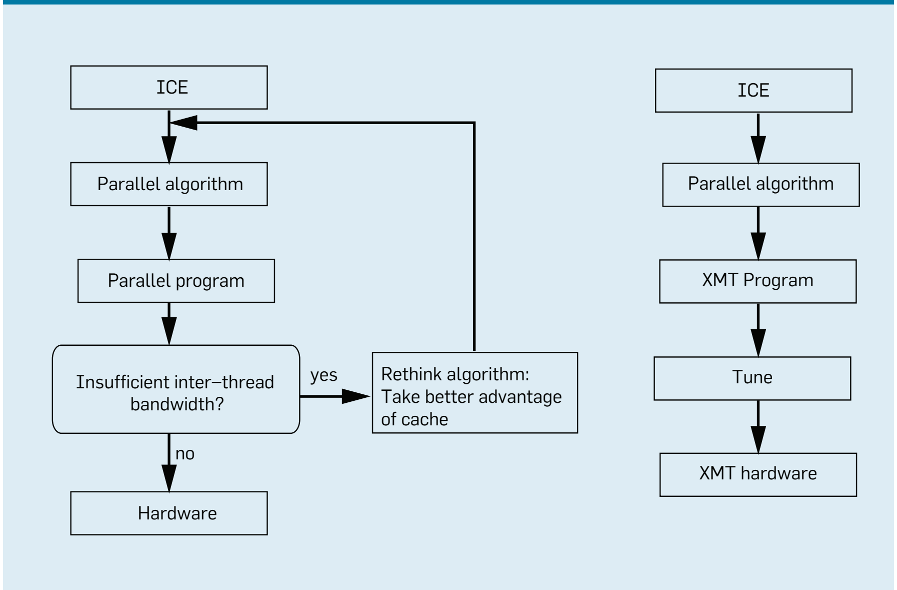


Figure 2. Right column is a workflow from an ICE abstraction of an algorithm to implementation; left column may never terminate.



a robust many-core platform coupled with a new many-core software spiral to serve the world of computing for years to come. A software spiral is basically an infrastructure for the economy. Since advancing infrastructures generally depends on government funding, designating software-spiral rebirth a killer app also motivates funding agencies and major vendors to support the work. The impact on manufacturing productivity could further motivate them.

Programmer Workflow

ICE requires the lowest level of cognition from the programmer relative to all current parallel programming models. Other approaches require additional steps (such as decomposition[10]). In CS theory, the speedup provided by parallelism is measured as work divided by depth; reducing the advantage of ICE/PRAM to practice is a different matter.

The reduction to practice I have led relies on the programmer's workflow, as outlined in the right side of Figure 2. Later, I briefly cover the parallel-algorithms stage. The step-by-step PRAM explication, or "data-parallel" instructions, represents a traditional tightly synchronous outlook on parallelism. Unfortunately, tight step-by-step synchrony is not a good match with technology, including its power constraints.

To appreciate the difficulty of implementing step-by-step synchrony in hardware, consider two examples: Memories based on long tightly synchronous pipelines of the type seen in Cray vector machines have long been out of favor among architects of high-performance computing; and processing memory requests takes from one to 400 clock cycles. Hardware must be made as flexible as possible to advance without unnecessary waiting for concurrent memory requests.

To underscore the importance of the bridge the XMT approach builds from the tightly synchronous PRAM to relaxed synchrony implementation, note three known limitations with power consumption of multi-core architectures: high power consumption of the wide communication buses needed to implement cache coherence; basic nm complexity of cache-coherence traffic (given n cores and m invalidations) and implied toll on inter-core bandwidth; and high power consumption needed for a tightly synchronous implementation in silicon in these designs. The XMT approach addresses all three by avoiding hardware-supported cache-coherence altogether and by significantly relaxing synchrony.

Workflow is important, as it guides the human-to-machine process of programming; see Figure 2 for two workflows. The non-XMT hardware implementation on the left side of the figure may require revisiting and changing the algorithm to fit bandwidth constraints among threads of the computation, a programming process that doesn't always yield an acceptable outcome. However, the XMT hardware allows a workflow (right side of the figure) that requires tuning only for performance; revisiting and possibly changing the algorithm is generally not needed. An optimizing compiler should be able to do its own tuning without programmer intervention, as in serial computing.

Most of the programming effort in traditional parallel programming (domain partitioning, load balancing) is generally of lesser importance for exploiting on-chip parallelism, where parallelism overhead can be kept low and processor-to-memory bandwidth high. This observation drove development of the XMT programming model and its implementation by my research team. XMT is intended to provide a simpler parallel programming model that efficiently exploits on-chip parallelism through multiple design elements.

The XMT architecture uses a high-bandwidth low-latency on-chip interconnection network to provide more uniform memory-access latencies. Other specialized XMT hardware primitives allow concurrent instantiation of as many threads as the number of available processors, a count that can reach into the thousands. Specifically, XMT can perform two main operations: forward (instantly) program instructions to all processors in the time required to forward the instructions (for one thread) to just one processor; and reallocate any number of processors that complete their jobs at the same time to new jobs (along with their instructions) in the time required to reallocate one processor. The high-bandwidth, low-latency interconnection network and low-overhead creation of many threads allow efficient support for the fine-grain parallelism used to hide memory latencies and a programming model for which locality is less an issue than in designs with less bandwidth. These mechanisms support dynamic load balancing, relieving programmers from having to directly assign work to processors. The programming model is simplified further by letting threads run to completion without synchronization (no busy-waits) and synchronizing access to shared data with prefix-sum (fetch-and-add type) instructions. These features result in a flexible programming style that accommodates the ICE abstraction and encourages program development for a range of applications.

The reinvention of computing for parallelism also requires pulling together a number of technical communities. My 2009 paper[26] sought to build a bridge to other architectures by casting the abstraction-centric vision of this article as a possible module in them, identifying a limited number of capabilities the module provides and suggesting a preferred embodiment of these capabilities using concrete "hardware hooks." If it is possible to augment a computer architecture through them (with hardware hooks or other means), the ICE abstraction and the programmer's workflow, in line with this article, can be supported. The only significant obstacle in today's multi-core architectures is their large cache-coherent local caches. Their limited scalability with respect to power gives vendors more reasons beyond an easier programming model to let go of this obstacle.

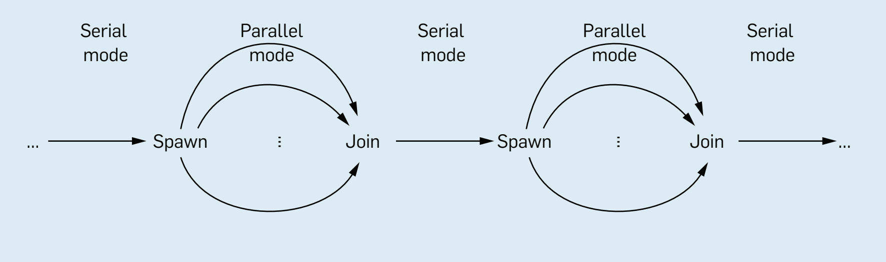
PRAM parallel algorithmic approach. The parallel random-access machine/model (PRAM) virtual model of computation is a generalization of the random-access machine (RAM) model.[9] RAM, the basic serial model behind standard programming languages, assumes any memory access or any operation (logic or arithmetic) takes unit-time (serial abstraction). The formal PRAM model assumes a certain number, say, p of processors, each able to concurrently access any location of a shared memory in the same time as a single access. PRAM has several submodels that differ by assumed outcome of concurrent access to the same memory location for either read or write purposes. Here, I note only one of them—the Arbitrary Concurrent-Read Concurrent-Write (CRCW) PRAM—which allows concurrent accesses to the same memory location for reads or writes; reads complete before writes, and an arbitrary write (to the same location, unknown in advance) succeeds. PRAM algorithms are essentially prescribed as a sequence of rounds and, for each round, up to p processors execute concurrently. The performance objective is to minimize the number of rounds. The PRAM parallel-algorithmic approach is well-known and has never been seriously challenged by any other parallel-algorithmic approach in terms of ease of thinking or wealth of knowledgebase. However, PRAM is also a strict formal model. A PRAM algorithm must therefore prescribe for each and every one of its p processors the instruction the processor executes at each time unit in a detailed computer-program-like fashion that can be quite demanding. The PRAM-algorithms theory mitigates this instruction-allocation scheme through the work-depth (WD) methodology.

This methodology (due to Shiloach and Vishkin[20]) suggests a simpler way to allocate instructions: A parallel algorithm can be prescribed as a sequence of rounds, and for each round, any number of operations can be executed concurrently, assuming unlimited hardware. The total number of operations is called "work," and the number of rounds is called "depth," as in the ICE abstraction. The first performance objective is to reduce work, and the immediate second one is to reduce depth. The methodology of restricting attention only to work and depth has been used as the main framework for the presentation of PRAM algorithms[16,17] and is in my class notes on the XMT home page http://www.umiacs.umd.edu/users/vishkin/XMT/. Deriving a full PRAM description from a WD description is easy. For concreteness, I demonstrate WD descriptions on two examples, the first concerning parallelism, the second concerning the WD methodology (see the sidebar "Parallel Algorithms").

The programmer's workflow starts with the easy-to-understand ICE abstraction and ends with the XMT system, providing a practical implementation of the vast PRAM algorithmic knowledge base.

XMT programming model. The programming model behind the XMT framework is an arbitrary concurrent read, concurrent write single program multiple data, or CRCW SPMD, programming model with two executing modes: serial and parallel. The two instructions—spawn and join—specify the beginning and end, respectively, of a parallel section (see Figure 3). An arbitrary number of virtual threads, initiated by a spawn and terminated by a join, share the same code. The workflow relies on the spawn command to extend the ICE abstraction from the WD methodology to XMT programming. As with the respective PRAM model, the arbitrary CRCW aspect dictates that concurrent writes to the same memory location result in an arbitrary write committing. No assumption needs to be made by the programmer beforehand about which one will succeed. An algorithm designed with this property in mind permits each thread to progress at its own speed, from initiating spawn to terminating join, without waiting for other threads—no thread "busy-waits" for another thread. The implied "independence of order semantics" allows XMT to have a shared memory with a relatively weak coherence model. An advantage of this easier-to-implement SPMD model is that it is PRAM-like. It also incorporates the prefix-sum statement operating on a base variable, B, and an increment variable, R. The result of a prefix-sum is that B gets the value B + R, while R gets the initial value of B, a result called "atomic" that's similar to fetch-and-increment in Gottlieb et al.[12]

Figure 3. Serial and parallel execution modes.



Merging with a Single Spawn-Join

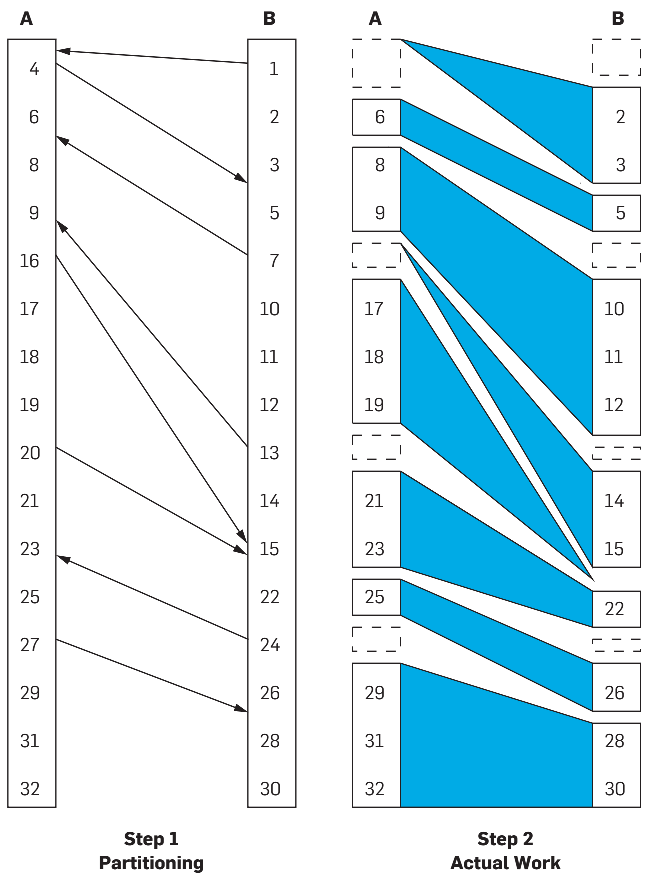
The merging problem takes as input two sorted arrays $A = A[1 \ldots n]$ and $B = B[1 \ldots n]$. Each of these $2n$ elements must then be mapped into an array $C = C[1 \ldots 2n]$ that is also sorted. I first review the Shiloach-Vishkin two-step PRAM algorithm for merging, then discuss its related XMTC programming:

Step 1. Partitioning. This step selects some number x of elements from A at equal distances. In the example in the figure here, suppose the $x = 4$ elements 4, 16, 20, and 27 are selected and ranked relative to array B using x concurrent binary searches. Similarly, x elements from B at equal distances, say, elements 1, 7, 13, and 24, are also selected, then ranked relative to array A using $x = 4$ concurrent binary searches. The step takes $O(\log n)$ time. These ranked elements partition the merging job that must be completed into $2x = 8$ "strips"; in the figure, step 2 includes eight such strips.

Step 2. Actual work. For each strip the remaining job is to merge a subarray of A with a subarray of B, mapping their elements into a subarray of C. Since these $2x$ merging jobs are mutually independent, each is able to concurrently apply the standard linear-time serial merging algorithm.

Consider the following complexity analysis of this algorithm: Since each strip has at most n/x elements from A and n/x elements from B, the depth (or parallel time) of the second step is $O(n/x)$. If $x \leq n/\log n$, the first step and the algorithm as a whole does $O(n)$ work. In the PRAM model, this algorithm requires $O(n/x + \log n)$ time. A simplistic XMTC program requires as many spawn (and respective join) commands as the number of PRAM steps. The reasons I include this example here are that it involves a way to use only a single spawn (and a single join) command to represent the whole merging algorithm and, as I explain in the Conclusion, to demonstrate an XMT advantage over current hardware by comparing it with the parallel merging algorithm in Cormen et al.[9]

Main steps of the ranking/merging algorithm.



Merging in XMTC. An XMTC program spawns $2x$ concurrent threads, one for each of the selected elements in array A or B. Using binary search, each thread first ranks its array element relative to the other array, then proceeds directly (without a join operation) to merge the elements in its strip, terminating just before setting the merging result of another selected element because the merging result is computed by another thread.

To demonstrate the operation of a thread, consider the thread of element 20. Starting with binary search on array B the thread finds that 20 ranks as 11 in B; 11 is the index of 15 in B. Since the index of 20 in A is 9, element 20 ranks 20 in C. The thread then compares 21 to 22 and ranks element 21 (as 21), then compares 23 to 22 to rank 22, 23 to 24 to rank 23, and 24 to 25 but terminates since the thread of 24 ranks 24, concluding the example.

Our experience is that, with little effort, XMT-type threading requires fewer synchronizations than implied by the original PRAM algorithm. The current merging example demonstrates that synchronization reduction is sometimes significant.

The primitive is especially useful when several threads perform a prefix-sum simultaneously against a common base, because multiple prefix-sum operations can be combined by the hardware to form a very fast multi-operand prefix-sum operation. Because each prefix-sum is atomic, each thread returns a different R value. This way, the parallel prefix-sum command can be used to implement efficient and scalable inter-thread synchronization by arbitrating an ordering among the threads.

The XMTC high-level language implements the programming model. XMTC is an extension of standard C, augmenting C with a small number of commands (such as spawn, join, and prefix-sum). Each parallel region is delineated by spawn and join statements, and synchronization is achieved through the prefix-sum and join commands. Every thread executing the parallel code is assigned a unique thread ID, designated $. The spawn statement takes as arguments the lowest ID and highest ID of the threads to be spawned. For the hardware implementation (discussed later), XMTC threads can be as short as eight to 10 machine instructions that are not difficult to get from PRAM algorithms. Programmers from high school to graduate school are pleasantly surprised by the flexibility of translating PRAM algorithms to XMTC multi-threaded programs. The ability to code the whole merging algorithm using a single spawn-join pair is one such surprise (see the sidebar "Merging with a Single Spawn-Join").

To demonstrate simple code, consider two code examples:

The first is a small XMTC program for the parallel exchange algorithm discussed in the "Parallel Algorithms" sidebar:

```
spawn ( 0 , n−1){
   var x
      x:=A( $ ) ;
     A( $ ):=B( $ ) ;
     B( $ ):=x
}
```

The program spawns a concurrent thread for each of the depth-3 serial-exchange iterations using a local variable x. Note that the join command is implied by the right parenthesis at the end of the program.

The second assumes an array of n integers A. The programmer wishes to "compact" the array by copying all non-zero values to another array, B, in an arbitrary order. The XMTC code is:

```
psBaseReg x=0;
spawn ( 0 , n−1){
        int e ;
        e=1;
        i f (A[ $ ] ) !=0) {
                    ps ( e , x ) ;
                    B[ e ]=A[ $ ]
}
}
```

It declares a variable x as the base value to be used in a prefix-sum command (ps in XMTC), initializing it to 0. It then spawns a thread for each of the n elements in A. A local thread variable e is initialized to 1. If the element of the thread is non-zero, the thread performs a prefix-sum to get a unique index into B where it can place its value.

Other XMTC commands. Prefix-sum-to-memory (psm) is another prefix-sum command, the base of which is any location in memory. While the increment of ps must be 0 or 1, the increment of psm is not limited, though its implementation is less efficient. Single Spawn (sspawn) is a command that can spawn an extra thread and be nested. A nested spawn command in XMTC code must be replaced (by programmer or compiler) by sspawn commands. The XMTC commands are described in the programmer's manual included in the software release on the XMT Web pages.

Tuning XMT programs for performance. My discussion here of performance tuning would be incomplete without a description of salient features of the XMT architecture and hardware. The XMT on-chip general-purpose computer architecture is aimed at the classic goal of reducing single-task completion time. The WD methodology gives algorithm designers the ability to express all the parallelism they observe. XMTC programming further permits expressing this virtual parallelism by letting programmers express as many concurrent threads as they wish. The XMT processor must now provide an effective way to map this virtual parallelism onto the hardware. The XMT architecture provides dynamic allocation of the XMTC threads onto the hardware for better load balancing. Since XMTC threads can be short, the XMT hardware must directly manage XMT threads to keep overhead low. In particular, an XMT program looks like a single thread to the operating system (see the sidebar "The XMT Processor" for an overview of XMT hardware).

The main thing performance programmers must know in order to tune the performance of their XMT programs is that a ready-to-run version of an XMT program depends on several parameters: the length of the (longest) sequence of roundtrips to memory (LSRTM); queuing delay to the same shared memory location (known as queue-read queue-write, or QRQW[11]); and work and depth. Their optimization is a responsibility shared subtly by the architecture, the compiler, and the programmer/algorithm designer.

See Vishkin et al[27] for a demonstration of tuning XMTC code for performance by accounting for LSRTM. As an example, it improves XMT hardware performance on the problem of summing n numbers.

Execution can differ from the literal XMTC code in order to keep the size of working space under control or otherwise improve performance. For example, compiler and runtime methods could perform this modification by clustering virtual threads offline or online and prioritize execution of nested spawns using known heuristics based on a mix of depth-first and breadth-first searches.

Commitments to silicon of XMT by my research team at the University of Maryland include a 64-processor, 75MHz computer based on field-programmable gate array (FPGA) technology developed by Wen[28] and 64-processor ASIC 10mm X 10mm chip using IBM's 90nm technology developed together by Balkan, Horak, Keceli, and Wen (see Figure 4). Tzannes and Cargaea (guided by Barua and me) have also developed a basic yet stable compiler, and Keceli has developed a cycle-accurate simulator of XMT. Both are available through the XMT software release on the XMT Web pages.

Easy to build. An individual graduate student with no prior design experience completed the XMT hardware description (in Verilog) in just over two years (2005–2007). XMT is also silicon-efficient. The ASIC design by the XMT research team at the University of Maryland shows that a 64-processor XMT needs the same silicon area as a (single) current commodity core. The XMT approach goes after any type of application parallelism regardless of how much parallelism the application requires, the regularity of this parallelism, or the parallelism's grain size, and is amenable to standard multiprogramming where the hardware supports several concurrent operating-system threads.

The XMT team has demonstrated good XMT performance, independent software engineers have demonstrated XMT programmability (see Hochstein et al.[14]), and independent education professionals have demonstrated

Eye-of-a-Needle Aphorism

Introduced at a time of hardware scarcity almost 70 years ago, the von Neumann apparatus of stored program and program counter forced the threading of instructions through a metaphoric eye of a needle. Coupling of mathematical induction and (serial) ISE abstraction was engineered to provide this threading, as discussed throughout the article. See especially the description of how variable X is used in the pseudo-code of the serial iterative algorithm in the exchange problem; also the first-in-first-out queue data structure in the serial breadth-first search; and the serial merging algorithm in which two elements are compared at a time, one from each of two sorted input arrays. As eye-of-a-needle threading is already second nature for many programmers, it has come to be associated with ease of programming.

Threading through the eye of a needle is an aphorism for extreme difficulty, even impossibility, in the broader culture, including in the texts of three major religions. The XMT extension to the von Neumann apparatus (noted in "The XMT Processor" sidebar) exploits today's relative abundance of hardware resources to free computing from the constraint of threading through the original apparatus. Coupling mathematical induction and the ICE abstraction explored here is engineered to capitalize on this freedom for ease of parallel programming and improved machine and application performance.

XMT teachability (see Torbert et al.[23]). Highlights include evidence of 100X speedups on general-purpose applications on a simulator of 1,000 on-chip processors[13] and speedups ranging from 15X to 22X for irregular problems (such as Quicksort, breadth-first search on graphs, finding the longest path in a directed acyclic graph), and speedups of 35X–45X for regular programs (such as matrix multiplication and convolution) on the 64-processor XMT prototype versus the best serial code on XMT.[28]

In 2009, Caragea et al.[8] demonstrated nearly 10X average performance improvement potential relative to Intel Core 2 Duo for a 64-processor XMT chip using the same silicon area as a single core. In 2010, Caragea et al.[7] demonstrated that, using the same silicon area as a modern graphics processing unit (GPU), the XMT design achieves an average speedup of 6X relative to the GPU for irregular applications and falls only slightly behind on regular ones. All GPU code was written and optimized by researchers and programmers unrelated to the XMT project.

With few exceptions, parallel programming approaches that dominated parallel computing prior to many-cores are still favored by vendors, as well as high-performance users. The steps they require include decomposition, assignments, orchestration, and mapping.[10] Indeed, parallel programming difficulties have failed all general-purpose parallel systems to date by limiting their use. In contrast, XMT frees its programmers from doing all the steps, in line with the ICE/PRAM abstraction.

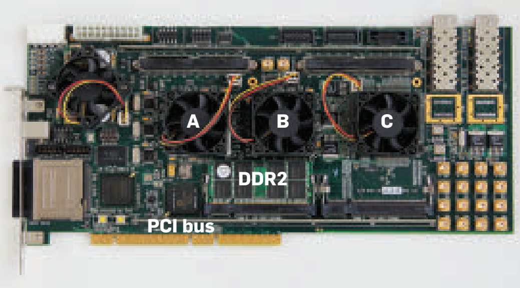
The XMT software environment release (a 2010 release of the XMTC compiler and cycle-accurate simulator of XMT) is available by free download from the XMT home page and from sourceforge.net, along with extensive documentation, and can be downloaded to any standard desktop computing platform. Teaching materials covering a class-tested programming methodology in which students are taught only parallel algorithms are also available from the XMT Web pages.

Most CS programs today graduate students to a job market certain to be dominated by parallelism but without the preparation they need. The level of awareness of parallelism required by the ICE/PRAM abstraction is so basic it is necessary for all other current approaches. As XMT is also buildable, the XMT approach is sufficient for programming a real machine. I therefore propose basing the introduction of the new generation of CS students to parallelism on the workflow presented here, at least until CS generally converges on a many-core platform.

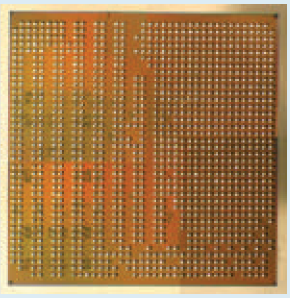
Related efforts. Related efforts toward parallelism come in several flavors; for example, Valiant's Multi-BSP bridging model for multi-core computing[24] appears closest to the XMT focus on abstraction. The main difference, however, is that XMT aims to preempt known shortcomings in existing machines by showing how to build machines differently, while the modeling in Valiant[24] aims to improve understanding of existing machines.

These prescriptive versus descriptive objectives are not the only difference. Valiant[24] modeled relatively low-level parameters of certain multi-core architectures, making them closer to Vishkin et al.[27] than to this article. Unlike both sources, simplicity drives the "one-liner" ICE abstraction. Parallel languages (such as CUDA, MPI, and OpenMP) tend to be different from computational models, as they often do not involve performance modeling. They require a level of detail that distances them farther from simple

Figure 4. Left side. FPGA board (size of a car license plate) with three FPGA chips (generously donated by Xilinx): A, B: Virtex-4LX200; C: Virtex-4FX100. Right side. 10mm X 10mm chip using IBM Flip-Chip technology.



A, B: Virtex-4LX200.
C: Virtex-4FX100.



10mm X 10mm chip using IBM Flip-Chip technology.

The XMT Processor

The XMT processor (see the figure here) includes a master thread control unit (MTCU), processing clusters, each comprising several thread-control units (TCUs), a high-bandwidth low-latency interconnection network[3] and its extension to a globally asynchronous locally synchronous, GALS-style, design incorporating asynchronous logic,[15,18] memory modules (MM), each comprising on-chip cache and off-chip memory, prefix-sum (PS) unit(s), and global registers. The shared-memory-modules block (bottom left of the figure) suppresses the sharing of a memory controller by several MMs. The processor alternates between serial mode (in which only the MTCU is active) and parallel mode. The MTCU has a standard private data cache (used in serial mode) and a standard instruction cache. The TCUs, which lack a write data cache, share the MMs with the MTCU.

The overall XMT design is guided by a general design ideal I call no-busy-wait finite-state-machines, or NBW FSM, meaning the FSMs, including processors, memories, functional units, and interconnection networks comprising the parallel machine, never cause one another to busy-wait. It is ideal because no parallel machine can operate that way. Nontrivial parallel processing demands the exchange of results among FSMs. The NBW FSM ideal represents my aspiration to minimize busy-waits among the various FSMs comprising a machine.

Here, I cite the example of how the MTCU orchestrates the TCUs to demonstrate the NBW FSM ideal. The MTCU is an advanced serial microprocessor that also executes XMT instructions (such as spawn and join). Typical program execution flow, as in Figure 3, can also be extended through nesting of sspawn commands. The MTCU uses the following XMT extension to the standard von Neumann apparatus of the program counters and stored program: Upon encountering a spawn command, the MTCU broadcasts the instructions in the parallel section starting with that spawn command and ending with a join command on a bus connecting to all TCU clusters.

The largest ID number of a thread the current spawn command must execute Y is also broadcast to all TCUs. The ID (index) of the largest executing threads is stored in a global register X. In parallel mode, a TCU executes one thread at a time. Executing a thread to completion (upon reaching a join command), the TCU does a prefix-sum using the PS unit to increment global register X. In response, the TCU gets the ID of the thread it could execute next; if the ID is ≤Y, the TCU executes a thread with this ID. Otherwise, the TCU reports to the MTCU that it finished executing. When all TCUs report they've finished, the MTCU continues in serial mode.

The broadcast operation is essential to the XMT ability to start all TCUs at once in the same time it takes to start one TCU. The PS unit allows allocation of new threads to the TCUs that just became available within the same time it takes to allocate one thread to one TCU. This dynamic allocation provides runtime load-balancing of threads coming from an XMTC program.

We are now ready to connect with the NBW FSM ideal. Consider an XMT program derived from the workflow. From the moment the MTCU starts executing a spawn command until each TCU terminates the threads allocated to it, no TCU can cause any other TCU to busy-wait for it. An unavoidable busy-wait ultimately occurs when a TCU terminates and begins waiting for the next spawn command.

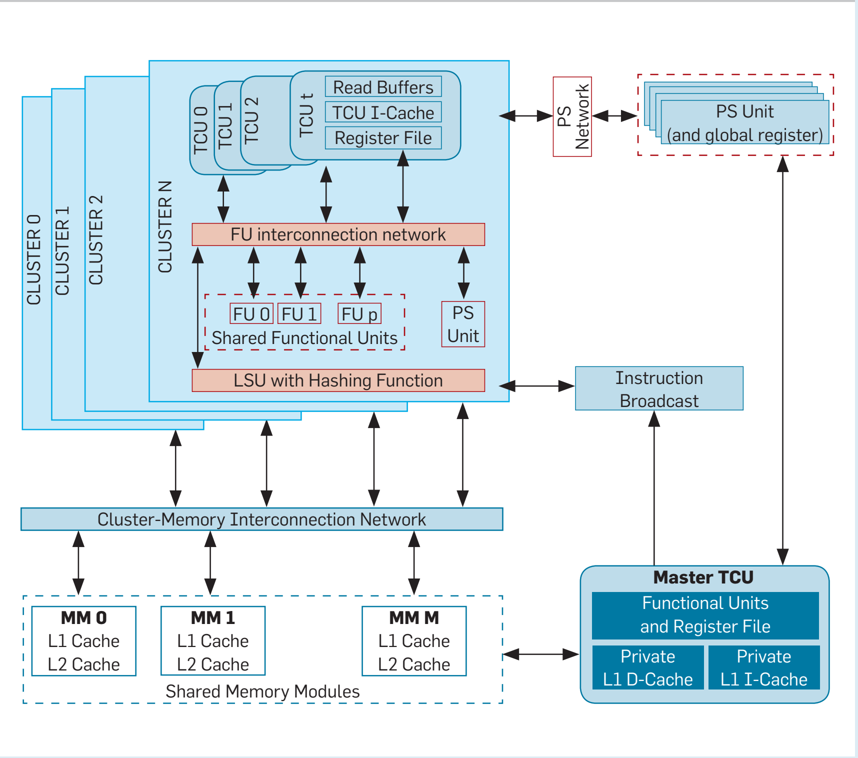
TCUs, with their own local registers, are simple in-order pipelines, including fetch, decode, execute/memory-access, and write-back stages. The FPGA computer has 64 TCUs in four clusters of 16 TCUs each. XMT designers and evangelists aspire to develop a machine with 1,024 TCUs in 64 clusters. A cluster includes functional units shared by several TCUs and one load/store port to the interconnection network shared by all its TCUs.

The global memory address space is evenly partitioned into the MMs through a form of hashing. The XMT design eliminates the cache-coherence problem, a challenge in terms of bandwidth and scalability. In principle, there are no local caches at the TCUs. Within each MM, the order of operations to the same memory location is preserved.

For performance enhancements (such as data prefetch) incorporated into the XMT hardware, along with more on the architecture, see Wen and Vishkin[28]; for more on compiler and runtime scheduling methods for nested parallelism, see Tzannes et al.,[23] and for prefetching methods, see Caragea et al.[6]

Patents supporting the XMT hardware were granted from 2002 to 2010, appearing in Nuzman and Vishkin[18] and Vishkin.[25]

Block diagram of the XMT architecture.



abstractions.

Several research centers[1,2] are actively exploring the general problems discussed here. The University of California, Berkeley, Parallel Computing Lab and Stanford University's Pervasive Parallelism Laboratory advocate an application-driven approach to reinventing computing for parallelism.

Conclusion

The vertical integration of a parallel-processing system, compiler, programming, and algorithms proposed here through the XMT framework with the ICE/PRAM abstraction as its front-end is notable for its relative simplicity. ICE is a newly defined feature that has not appeared in prior research, including my own, and is more rudimentary than prior parallel computing concepts. Rudimentary concepts are the basis for the fundamental development of any field. ICE can be viewed as an axiom that builds on mathematical induction, one of the more rudimentary concepts in mathematics. The suggestion here of using a simple abstraction as the guiding principle for reinventing computing for parallelism also appears to be new. Considerable evidence suggests it can be done (see the sidebar "Eye-of-a-Needle Aphorism").

The following comparison with a chapter on multithreading algorithms in the 2009 textbook *Introduction to Algorithms* by Cormen et al.[9] helps clarify some of the article's contributions. The 1990 first edition of Cormen et al.[9] included a chapter on PRAM algorithms emphasizing the role of work-depth design and analysis; the 2009 chapter[9] likewise emphasized work-depth analysis. However, to match current commercial hardware, the 2009 chapter turned to a variant of dynamic multithreading (in lieu of work-depth design) in which the main primitive was similar to the XMT sspawn command (discussed here). One thread was able to generate only one more thread at a time; these two threads would then generate one more thread each, and so on, instead of freeing the programmer to directly design for the work-depth analysis that follows (per the same 2009 chapter).

The XMT on-chip general-purpose computer architecture is aimed at the classic goal of reducing single-task completion time.

Cormen et al.'s[9] dynamic multithreading should encourage hardware enhancement to allow simultaneously starting many threads in the same time required to start one thread. A step ahead of available hardware, XMT includes a spawn command that spawns any number of threads upon transition to parallel mode. Moreover, the ICE abstraction incorporates work-depth early in the design workflow, similar to Cormen et al.'s 1990 first edition.[9]

The $O(\log n)$ depth parallel merging algorithm versus the $O(\log^2 n)$ depth one in Cormen et al.[9] demonstrated an XMT advantage over current hardware, as XMT allows a parallel algorithm for the same problem that is both faster and simpler. The XMT hardware scheduling brought the hardware performance model much closer to work-depth and allowed the XMT workflow to streamline the design with the analysis from the start.

Several features of the serial paradigm made it a success, including a simple abstraction at the heart of the "contract" between programmers and builders, the software spiral, ease of programming, ease of teaching, and backward compatibility on serial code and application programming. The only feature that XMT, as in other multi-core approaches, does not provide is speedups for serial code. The ICE/PRAM/XMT workflow and architecture provide a viable option for the many-core era. My XMT solution should challenge and inspire others to come up with competing abstraction proposals or alternative architectures for ICE/PRAM. Consensus around an abstraction will move CS closer to convergence toward a many-core platform and putting the software spiral back on track.

The XMT workflow also gives programmers a productivity advantage. For example, I have traced several errors in student-developed XMTC programs to shortcuts the students took around the ICE algorithms. Overall, improved understanding of programmer productivity, a traditionally difficult issue in parallel computing, must be a top priority for architecture research. To the extent possible, evaluation of productivity should be on par with performance and power. For starters, productivity benchmarks must be developed.

Ease of programming, or programmability, is a necessary condition for the success of any many-core platform, and teachability is a necessary condition for programmability and in turn for productivity. The teachability of the XMT approach has been demonstrated extensively; for example, since 2007 more than 100 students in grades K–12 have learned to program XMT, including in two magnet programs: Montgomery Blair High School, Silver Spring, MD, and Thomas Jefferson High School for Science and Technology, Alexandria, VA.[22] Others are Baltimore Polytechnic High School, where 70% of the students are African American, and a summer workshop for middle-school students from underrepresented groups in Montgomery County, MD, public schools.

In the fall of 2010, I jointly conducted another experiment, this one via video teleconferencing with Professor David Padua of the University of Illinois, Urbana-Champaign using Open MP and XMTC, with XMTC programming assignments run on the XMT 64-processor FPGA machine. Our hope was to produce a meaningful comparison of programming development time from the 30 participating Illinois students. The topics and problems covered in the PRAM/XMT part of the course were significantly more advanced than Open MP alone. Having sought to demonstrate the importance of teachability from middle school on up, I strongly recommend that it becomes a standard benchmark for evaluating many-core hardware platforms.

Blake et al.[4] reported that after analyzing current desktop/laptop applications for which the goal was better performance, the applications tend to comprise many threads, though few of them are used concurrently; consequently, the applications fail to translate the increasing thread-level parallelism in hardware to performance gains. This problem is not surprising given that most programmers can't handle multi-core microprocessors. In contrast, guided by the simple ICE abstraction and by the rich PRAM knowledgebase to find parallelism, XMT programmers are able to represent that parallelism using a type of threading the XMT hardware is engineered to exploit for performance.

For more, see the XMT home page at the University of Maryland http://www.umiacs.umd.edu/users/vishkin/XMT/. The XMT software environment release is available by free download there and from sourceforge.net at http://sourceforge.net/projects/xmtc/, along with extensive documentation. A 2010 release of the XMTC compiler and cycle-accurate simulator of XMT can also be downloaded to any standard desktop computing platform. Teaching materials covering a University of Maryland class-tested programming methodology in which even college freshmen and high school students are taught only parallel algorithms are also available from the XMT Web pages.

Acknowledgment

This work is supported by the National Science Foundation under grant 0325393.

References

1. Adve, S. et al. *Parallel Computing Research at Illinois: The UPCRC Agenda.* White Paper. University of Illinois, Champaign-Urbana, IL,2008; http://www.upcrc.illinois.edu/UPCRC_Whitepaper.pdf
2. Asanovic, K. et al. *The Landscape of Parallel Computing Research: A View from Berkeley.* Technical Report UCB/EECS-2006-183. University of California, Berkeley, 2006; http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf
3. Balkan, A., Horak, M., Qu, G., and Vishkin, U. Layout-accurate design and implementation of a high-throughput interconnection network for single-chip parallel processing. In *Proceedings of the 15th Annual IEEE Symposium on High Performance Interconnects* (Stanford, CA, Aug. 22–24). IEEE Press, Los Alamitos, CA, 2007.
4. Blake, G., Dreslinski, R., Flautner, K., and Mudge, T. Evolution of thread-level parallelism in desktop applications. In *Proceedings of the 37th Annual International Symposium on Computer Architecture* (Saint-Malo, France, June 19–23). ACM Press, New York, 2010, 302–313.
5. Borkar, S. et al. *Platform 2015: Intel Processor and Platform Evolution for the Next Decade. White Paper.* Intel, Santa Clara, CA, 2005; http://epic.hpi.uni-potsdam.de/pub/Home/TrendsAndConceptsII2010/HW_Trends_borkar_2015.pdf
6. Caragea, G., Tzannes, A., Keceli, F., Barua, R., and Vishkin, U. Resource-aware compiler prefetching for many-cores. In *Proceedings of the Ninth International Symposium on Parallel and Distributed Computing* (Istanbul, Turkey, July 7–9). IEEE Press, Los Alamitos, CA, 2010, 133–140.
7. Caragea, G., Keceli, F., Tzannes, A., and Vishkin, U. General-purpose vs. GPU: Comparison of many-cores on irregular workloads. In *Proceedings of the Second Usenix Workshop on Hot Topics in Parallelism* (University of California, Berkeley, June 14–15). Usenix, Berkeley, CA, 2010.
8. Caragea, G., Saybasili, B., Wen, X., and Vishkin, U. Performance potential of an easy-to-program PRAM-on-chip prototype versus state-of-the-art processor. In *Proceedings of the 21st ACM SPAA Symposium on Parallelism in Algorithms and Architectures* (Calgary, Canada, Aug. 11–13). ACM Press, New York, 2009, 163–165.
9. Cormen, T., Leiserson, C., Rivest, R., and Stein, C. *Introduction to Algorithms, Third Edition.* MIT Press, Cambridge, MA, 2009.
10. Culler, D. and Singh, J. *Parallel Computer Architecture: A Hardware/Software Approach.* Morgan-Kaufmann, San Francisco, CA, 1999.
11. Gibbons, P., Matias, Y., and Ramachandran, V. The queue-read queue-write asynchronous PRAM model. *Theoretical Computer Science 196*, 1–2 (Apr. 1998), 3–29.
12. Gottlieb, A. et al. The NYU ultracomputer designing an MIMD shared-memory parallel computer. *IEEE Transactions on Computers 32*, 2 (Feb. 1983), 175–189.
13. Gu, P. and Vishkin, U. Case study of gate-level logic simulation on an extremely fine-grained chip multiprocessor. *Journal of Embedded Computing 2*, 2 (Apr. 2006), 181–190.
14. Hochstein, L., Basili, V., Vishkin, U., and Gilbert, J. A pilot study to compare programming effort for two parallel programming models. *Journal of Systems and Software 81*, 11 (Nov. 2008), 1920–1930.
15. Horak, M., Nowick, S., Carlberg, M., and Vishkin, U. A low-overhead asynchronous interconnection network for gals chip multiprocessor. In *Proceedings of the Fourth ACM/IEEE International Symposium on Networks-on-Chip* (Grenoble, France, May 3–6). IEEE Computer Society, Washington D.C., 2010, 43–50.
16. JaJa, J. *An Introduction to Parallel Algorithms.* Addison-Wesley Publishing Company, Reading, MA, 1992.
17. Keller, J., Kessler, C., and Traeff, J. *Practical PRAM Programming.* Wiley-Interscience, New York, 2001.
18. Nuzman, J. and Vishkin, U. *Circuit Architecture for Reduced-Synchrony-On-Chip Interconnect.* U.S. Patent 6,768,336, 2004; http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&p=1&u=%2Fnetahtml%2FPTO%2Fsearch-bool.html&r=1&f=G&l=50&co1=AND&d=PTXT&s1=6768336.PN.&OS=PN/6768336&RS=PN/6768336
19. Patterson, D. The trouble with multi-core: Chipmakers are busy designing microprocessors that most programmers can't handle. *IEEE Spectrum* (July 2010).
20. Shiloach, Y. and Vishkin, U. An $O(n^2 \log n)$ parallel max-flow algorithm. *Journal of Algorithms 3*, 2 (Feb.1982), 128–146.
21. Sutter, H. The free lunch is over: A fundamental shift towards concurrency in software. *Dr. Dobb's Journal 30*, 3 (Mar. 2005).
22. Torbert, S., Vishkin, U., Tzur, R., and Ellison, D. Is teaching parallel algorithmic thinking to high school students possible? One teacher's experience. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, WI, Mar. 10–13). ACM Press, New York, 2010, 290–294.
23. Tzannes, A., Caragea, G., Barua, R., and Vishkin, U. Lazy binary splitting: A run-time adaptive dynamic works-stealing scheduler. In *Proceedings of the15th ACM Symposium on Principles and Practice of Parallel Programming* (Bangalore, India, Jan. 9–14). ACM Press, New York, 2010, 179–189.
24. Valiant, L. A bridging model for multi-core computing. In *Proceedings of the European Symposium on Algorithms* (Karlruhe, Germany, Sept. 15–17). Lecture Notes in Computer Science 5193. Springer, Berlin, 2008, 13–28.
25. Vishkin, U. U.S. Patents 6,463,527; 6,542,918; 7,505,822; 7,523,293; 7,707,388, 2002–2010; http://patft.uspto.gov/
26. Vishkin, U. Algorithmic approach to designing an easy-to-program system: Can it lead to a hardware-enhanced programmer's workflow add-on? In *Proceedings of the 27th International Conference on Computer Design* (Lake Tahoe, CA, Oct. 4–7). IEEE Computer Society, Washington D.C., 2009, 60–63.
27. Vishkin, U., Caragea, G., and Lee, B. Models for advancing PRAM and other algorithms into parallel programs for a PRAM-on-chip platform. In *Handbook on Parallel Computing*, S. Rajasekaran and J. Reif, Eds. Chapman and Hall/CRC Press, Boca Raton, FL, 2008, 5.1-60.
28. Wen, X. and Vishkin, U. FPGA-based prototype of a PRAM-on-chip processor. In *Proceedings of the Fifth ACM Conference on Computing Frontiers* (Ischia, Italy, May 5–7). ACM Press, New York, 2008, 55–66.

Uzi Vishkin (vishkin@umd.edu) is a professor in the University of Maryland Institute for Advanced Computer Studies (http://www.umiacs.umd.edu/~vishkin) and Electrical and Computer Engineering Department, College Park, MD.

DOI:10.1145/1866739.1866758

What does it mean to preserve privacy?

BY CYNTHIA DWORK

A Firm Foundation for Private Data Analysis

IN THE INFORMATION realm, loss of privacy is usually associated with failure to control access to information, to control the flow of information, or to control the purposes for which information is employed. Differential privacy arose in a context in which ensuring privacy is a challenge even if all these control problems are solved: privacy-preserving statistical analysis of data.

The problem of *statistical disclosure control*—revealing accurate statistics about a set of respondents while preserving the privacy of individuals—has a venerable history, with an extensive literature spanning statistics, theoretical computer science, security, databases, and cryptography (see, for example, the excellent survey of Adam and Wortmann,[1] the discussion of related work in Blum et al.,[2] and the *Journal of Official Statistics* dedicated to confidentiality and disclosure control).

This long history is a testament to the importance of the problem. Statistical databases can be of enormous social value; they are used for apportioning resources, evaluating medical therapies, understanding the spread of disease, improving economic utility, and informing us about ourselves as a species.

The data may be obtained in diverse ways. Some data, such as census, tax, and other sorts of official data, is compelled; other data is collected opportunistically, for example, from traffic on the Internet, transactions on Amazon, and search engine query logs; other data is provided altruistically, by respondents who hope that sharing their information will help others to avoid a specific misfortune, or more generally, to increase the public good. Altruistic data donors are typically promised their individual data will be kept confidential—in short, they are promised "privacy." Similarly, medical data and legally compelled data, such as census data and tax return data, have legal privacy

» key insights

- **In analyzing private data, only by focusing on rigorous privacy guarantees can we convert the cycle of "propose-break-propose again" into a path of progress.**
- **A natural approach to defining privacy is to require that accessing the database teaches the analyst nothing about any individual. But this is problematic: the whole point of a statistical database is to teach general truths, for example, that smoking causes cancer. Learning this fact teaches the data analyst something about the likelihood with which certain individuals, not necessarily in the database, will develop cancer. We therefore need a definition that separates the utility of the database (learning that smoking causes cancer) from the increased risk of harm due to joining the database. This is the intuition behind *differential privacy*.**
- **This can be achieved, often with low distortion. The key idea is to randomize responses so as to effectively hide the presence or absence of the data of any individual *over the course of the lifetime of the database*.**

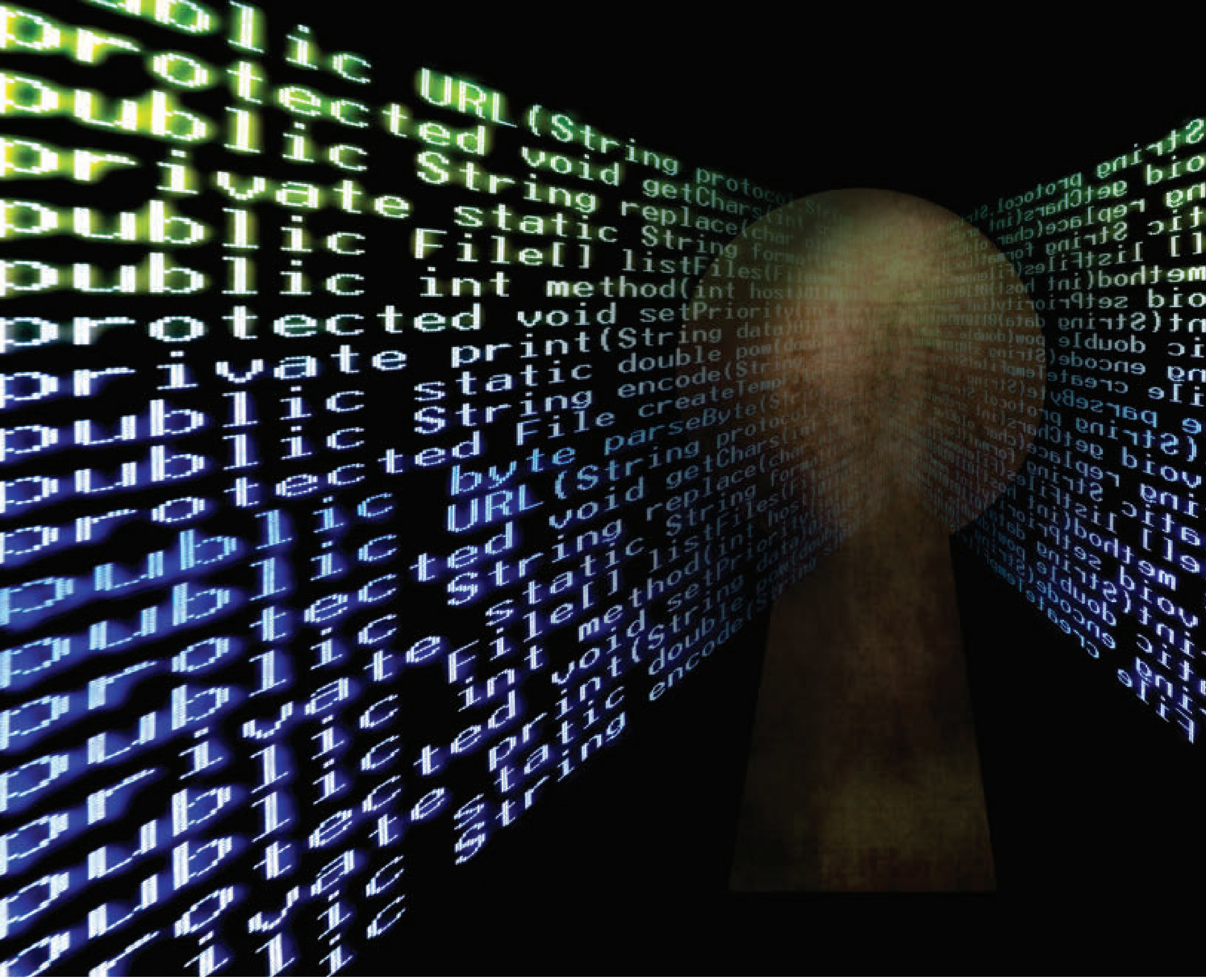


IMAGE BY BRIAN GREENBERG

mandates. In my view, ethics demand that opportunistically obtained data should be treated no differently, especially when there is no reasonable alternative to engaging in the actions that generate the data in question.

The problems remain: even if data encryption, key management, access control, and the motives of the data curator are all unimpeachable, what does it mean to preserve privacy, and how can it be accomplished?

"How" Is Hard

Let us consider a few common suggestions and some of the difficulties they can encounter.

Large Query Sets. One frequent suggestion is to disallow queries about a specific individual or small set of individuals. A well-known differencing argument demonstrates the inadequacy of the suggestion. Suppose it is known that Mr. X is in a certain medical database. Taken together, the answers to the two large queries "How many people in the database have the sickle cell trait?" and "How many people, not named X, in the database have the sickle cell trait?" yield the sickle cell status of Mr. X. The example also shows that encrypting the data, another frequent suggestion (oddly), would be of no help at all. The privacy compromise arises from correct operation of the database.

In *query auditing*, each query to the database is evaluated in the context of the query history to determine if a response would be disclosive; if so, then the query is refused. For example, query auditing might be used to interdict the pair of queries about sickle cell trait just described. This approach is problematic for several reasons, among them that query monitoring is computationally infeasible[16] and that the refusal to respond to a query may itself be disclosive.[15]

We think of a database as a collection of *rows*, with each row containing the data of a different respondent. In *subsampling* a subset of the rows is chosen at random and released. Statistics can then be computed on the subsample and, if the subsample is sufficiently large, these may be representative of the dataset as a whole. If the size of the subsample is very small compared to the size of the dataset, this approach has the property that every respondent is unlikely to appear in the subsample. However, this is clearly insufficient: Suppose appearing in a subsample has terrible consequences. Then every time subsampling occurs *some* individual suffers horribly.

In *input perturbation*, either the data or the queries are modified before a response is generated. This broad category encompasses a generalization of subsampling, in which the curator first chooses, based on a secret, random, function of the query, a subsample from the database, and then returns the result obtained by applying the query to the subsample.[4] A nice feature of this approach is that repeating the same query yields the same answer, while semantically equivalent but syntactially different queries are made on essentially unrelated subsamples. However, an outlier may only be protected by the unlikelihood of being in the subsample.

In what is traditionally called *randomized response*, the data itself is randomized once and for all and statistics are computed from the noisy responses, taking into account the distribution on the perturbation.[23] The term "randomized response" comes from the practice of having the respondents to a survey flip a coin and, based on the outcome, answering an invasive yes/no question or answering a more emotionally neutral one. In the computer science literature the choice governed by the coin flip is usually between honestly reporting one's value and responding randomly, typically by flipping a second coin and reporting the outcome. Randomized response was devised for the setting in which the individuals do not trust the curator, so we can think of the randomized responses as simply being published. Privacy comes from the uncertainty of how to interpret a reported value. The approach becomes untenable for complex data.

Adding random noise to the output has promise, and we will return to it later; here we point out that if done naïvely this approach will fail. To see this, suppose the noise has mean zero and that fresh randomness is used in generating every response. In this case, if the same query is asked repeatedly, then the responses can be averaged, and the true answer will eventually emerge. This is disastrous: an adversarial analyst could exploit this to carry out the difference attack described above. The approach cannot be "fixed" by recording each query and providing the same response each time a query is re-issued. There are several reasons for this. For example, syntactically different queries may be semantically equivalent, and if the query language is sufficiently rich, then the equivalence problem itself is undecidable, so the curator cannot even test for this.

Problems with noise addition arise even when successive queries are completely unrelated to previous queries.[5] Let us assume for simplicity that the database consists of a single—but very sensitive—bit per person, so we can think of the database as an n-bit Boolean vector $d = (d_1, \ldots, d_n)$. This is an abstraction of a setting in which the database rows are quite complex, for example, they may be medical records, but the attacker is interested in one specific field, such as HIV status. The abstracted attack consists of issuing a string of queries, each described by a subset S of the database rows. The query is asking how many 1's are in the selected rows. Representing the query as the n-bit characteristic vector $\mathbf{S}$ of the set S, with 1's in all the positions corresponding to rows in S and 0's everywhere else; the true answer to the query is the inner product $A(S) = \sum_{i=1}^{n} d_i \boldsymbol{S}_i$. Suppose the privacy mechanism responds with $A(S)$ + random noise. How much noise is needed in order to preserve privacy?

Since we have not yet defined privacy, let us consider the easier problem of avoiding blatant "non-privacy," defined as follows: the system is blatantly non-private if an adversary can construct a candidate database that agrees with the real database D in, say, 99% of the entries. An easy consequence of the following theorem is that a privacy mechanism adding noise with magnitude always bounded by, say, $n/401$ is blatantly non-private against an adversary that can ask all 2^n possible queries.[5] There is nothing special about 401; any number exceeding 400 would work.

Theorem 1. *Let $\mathcal{M}$ be a mechanism that adds noise bounded by E. Then there exists an adversary that can reconstruct the database to within $4E$ positions.*[5]

Blatant non-privacy with $E = n/401$ follows immediately from the theorem, as the reconstruction will be accurate in all but at most $4E = n \cdot \frac{4}{401} < n/100$ positions.

Proof. Let d be the true database. The adversary can attack in two phases:

1. **Estimate the number of 1's in all possible sets:** Query $\mathcal{M}$ on all subsets $S \subseteq [n]$.
2. **Rule out "distant" databases:** For every candidate database $c \in \{0, 1\}^n$, if, for any $S \subseteq [n]$, $\left|\sum_{i \in S} c_i - \mathcal{M}(S)\right| > E$, then rule out c. If c is not ruled out, then output c and halt.

Since $\mathcal{M}(S)$ never errs by more than E, the real database will not be ruled out, so this simple (but inefficient!) algorithm will output *some* database; let us call it c. We will argue that the number of positions in which c and d differ is at most $4 \cdot E$.

Let I_0 be the indices in which $d_i = 0$, that is, $I_0 = \{i \mid d_i = 0\}$. Similarly, define $I_1 = \{i \mid d_i = 1\}$. Since c was not ruled out, $|\mathcal{M}(I_0) - \sum_{i \in I_0} c_i| \leq E$. However, by assumption $|\mathcal{M}(I_0) - \sum_{i \in I_0} d_i| \leq E$. It follows from the triangle inequality that c and d differ in at most $2E$ positions in I_0; the same argument shows that they differ in at most $2E$ positions in I_1. Thus, c and d agree on all but at most $4E$ positions. □

What if we consider more realistic bounds on the number of queries? We think of $\sqrt{n}$ as an interesting threshold on noise, for the following reason: If the database contains n people drawn uniformly at random from a population of size $N \gg n$, and the fraction of the population satisfying a given condition is p, then we expect the number of rows in the database satisfying p to be roughly $np \pm \Theta\left(\sqrt{n}\right)$, by the properties of the hypergeometric distribution. That is, the sampling error is on the order of $\sqrt{n}$. We would like that the noise introduced for privacy is smaller than the sampling error, ideally $o\left(\sqrt{n}\right)$. Unfortunately, noise of magnitude $o\left(\sqrt{n}\right)$ is blatantly non-private against a series of $n \log^2 n$ *randomly generated* queries,[5] no matter the distribution on the noise. Several strengthenings of this pioneering result are now known. For example, if the entries in S are chosen independently according to a standard normal distribution, then blatant non-privacy continues to hold even against an adversary asking only $\Theta(n)$ questions, and even if more than a fifth of the responses have arbitrarily wild noise magnitudes, provided the other responses have noise magnitude $o(\sqrt{n})$.[8]

These are not just interesting mathematical exercises. We have been focusing on *interactive* privacy mechanisms, distinguished by the involvement of the curator in answering each query. In the *noninteractive* setting the curator publishes some information of arbitrary form, and the data is not used further. Research statisticians like to "look at the data," and we have frequently been asked for a method of generating a "noisy table" that will permit highly accurate answers to be derived for computations that are not specified at the outset. The noise bounds say this is impossible: No such table can safely provide very accurate answers to too many weighted subset sum questions; otherwise the table could be used in a simulation of the interactive mechanism, and an attack could be mounted against the table. Thus, even if the analyst only requires the responses to a small number of unspecified queries, the fact that the table can be exploited to gain answers to other queries is problematic.

In the case of "Internet scale" datasets, obtaining responses to, say, $n \geq 10^8$ queries is infeasible. What happens if the curator permits only a sublinear number of questions? This inquiry led to the first algorithmic results in differential privacy, in which it was shown how to maintain privacy against a sublinear number of *counting* queries, that is, queries of the form "How many rows in the database satisfy property P?" by adding noise of order $o(\sqrt{n})$—less than the sampling error —to each answer.[12] The cumbersome privacy guarantee, which focused on the question of what an adversary can learn about a row in the database, is now known to imply a natural and still very powerful relaxation of differential privacy, defined here.

"What" Is Hard

Newspaper horror stories about "anonymized" and "de-identified" data typically refer to noninteractive approaches in which certain kinds of information in each data record have been suppressed or altered. A famous example is AOL's release of a set of "anonymized" search query logs. People search for many "obviously" disclosive things, such as their full names (vanity searches), their own social security numbers (to see if their numbers are publicly available on the Web, possibly with a goal of assessing the threat of identity theft), and even the combination of mother's maiden name and social security number. AOL carefully redacted such obviously disclosive "personally identifiable information," and each user id was replaced by a random string. However, search histories can be very idiosyncratic, and a *New York Times* reporter correctly traced such an "anonymized" search history to a specific resident of Georgia.

It has taken several years to fully appreciate the importance of taking auxiliary information into account in privacy-preserving data release.

In a *linkage attack*, released data are linked to other databases or other sources of information. We use the term *auxiliary information* to capture information about the respondents *other* than that which is obtained through the (interactive or noninteractive) statistical database. Any priors, beliefs, or information from newspapers, labor statistics, and so on, all fall into this category.

In a notable demonstration of the power of auxiliary information, medical records of the governor of Massachusetts were identified by linking voter registration records to "anonymized" Massachusetts Group Insurance Commission (GIC) medical encounter data, which retained the birthdate, sex, and zip code of the patient.[22]

Despite this exemplary work, it has taken several years to fully appreciate the importance of taking auxiliary information into account in privacy-preserving data release. Sources and uses of auxiliary information are endlessly varied. As a final example, it has been proposed to modify search query logs by mapping *all* terms, not just the user ids, to random strings. In *token-based hashing* each query is tokenized, and then an uninvertible hash function is applied to each token. The intuition is that the hashes completely obscure the terms in the query. However, using a statistical analysis of the hashed log and *any* (unhashed) query log, for example, the released AOL log discussed above, the anonymization can be severely compromised, showing that token-based hashing is unsuitable for anonymization.[17]

As we will see next, there are deep reasons for the fact that auxiliary information plays such a prominent role in these examples.

Dalenius's Desideratum

In 1977 the statistician Tore Dalenius articulated an "*ad omnia*" (as opposed to *ad hoc*) privacy goal for statistical databases: Anything that can be learned about a respondent from the statistical database should be learnable without access to the database. Although informal, this feels like the "right" direction. The breadth of the goal captures all the common intuitions for privacy. In addition, the definition only holds the database accountable for whatever "extra" is learned about an individual, beyond that which can be learned from other sources. In particular, an extrovert who posts personal information on the Web may destroy his or her own privacy, and the database should not be held accountable.

Formalized, Dalenius' goal is strikingly similar to the gold standard for security of a cryptosystem against a passive eavesdropper, defined five years later. *Semantic security* captures the intuition that the encryption of a message reveals no information about the message. This is formalized by comparing the ability of a computationally efficient adversary, having access to both the ciphertext and any auxiliary information, to output (anything about) the plaintext, to the ability of a computationally efficient party having access *only* to the auxiliary information (and not the ciphertext), to achieve the same goal.[13] Abilities are measured by probabilities of success, where the probability space is over the random choices made in choosing the encryption keys, the ciphertexts, and by the adversaries. Clearly, if this difference is very, very tiny, then in a rigorous sense the ciphertext leaks (almost) no information about the plaintext.

The formal definition of semantic security is a pillar of modern cryptography. It is therefore natural to ask whether a similar property, such as Dalenius' goal, can be achieved for statistical databases. But there is an essential difference in the two problems. Unlike the eavesdropper on a conversation, the statistical database attacker is also a user, that is, a legitimate

The formal definition of semantic security is a pillar of modern cryptography. It is therefore natural to ask whether a similar property, such as Dalenius' goal, can be achieved for statistical databases.

consumer of the information provided by the statistical database (not to mention the fact that he or she may also be a respondent in the database).

Many papers in the literature attempt to formalize Dalenius' goal (in some cases unknowingly) by requiring that the adversary's prior and posterior beliefs about an individual (that is, before and after having access to the statistical database) should not be "too different," or that access to the statistical database should not change the adversary's views about any individual "too much." The difficulty with this approach is that if the statistical database teaches us anything at all, then it *should* change our beliefs about individuals. For example, suppose the adversary's (incorrect) prior view is that everyone has two left feet. Access to the statistical database teaches that almost everyone has one left foot and one right foot. The adversary now has a very different view of whether or not any given respondent has two left feet. But has privacy been compromised?

The last hopes for Dalenius' goal evaporate in light of the following parable, which again involves auxiliary information. Suppose we have a statistical database that teaches average heights of population subgroups, and suppose further that it is infeasible to learn this information (perhaps for financial reasons) in any other way (say, by conducting a new study). Finally, suppose that one's true height is considered sensitive. Given the auxiliary information "Turing is two inches taller than the average Lithuanian woman," access to the statistical database teaches Turing's height. In contrast, anyone without access to the database, knowing only the auxiliary information, learns much less about Turing's height.

A rigorous impossibility result generalizes this argument, extending to essentially any notion of privacy compromise, *assuming the statistical database is useful*. The heart of the attack uses extracted randomness from the statistical database as a one-time pad for conveying the privacy compromise to the adversary/user.[6,9]

Turing did not have to be a member of the database for the attack described earlier to be prosecuted against him. More generally, the things that statistical databases are designed to teach can, sometimes indirectly, cause damage to an individual, even if this individual is not in the database.

In practice, statistical databases are (typically) created to provide some anticipated social gain; they teach us something we could not (easily) learn without the database. Together with the attack against Turing, and the fact that he did not have to be a member of the database for the attack to work, this suggests a new privacy goal: Minimize the increased risk to an individual incurred by joining (or leaving) the database. That is, we move from comparing an adversary's prior and posterior views of an individual to comparing the risk to an individual when included in, versus when not included in, the database. This makes sense. A privacy guarantee that limits risk incurred by joining encourages participation in the dataset, increasing social utility. This is the starting point on our path to *differential privacy*.

Differential Privacy

Differential privacy will ensure that the ability of an adversary to inflict harm (or good, for that matter)—of any sort, to any set of people—should be essentially the same, independent of whether any individual opts in to, or opts out of, the dataset. We will do this indirectly, simultaneously addressing all possible forms of harm and good, by focusing on the probability of any given output of a privacy mechanism and how this probability can change with the addition or deletion of any row. Thus, we will concentrate on pairs of databases (D, D') differing only in one row, meaning one is a subset of the other and the larger database contains just one additional row. Finally, to handle worst-case pairs of databases, our probabilities will be over the random choices made by the privacy mechanism.

Definition 1. *A randomized function* $\mathcal{K}$ *gives* ε-differential privacy *if for all datasets D and D′ differing on at most one row, and all* $S \subseteq Range(\mathcal{K})$,

$$\Pr[\mathcal{K}(D) \in S] \leq \exp(\varepsilon) \times \Pr[\mathcal{K}(D') \in S], \quad (1)$$

where the probability space in each case is over the coin flips of $\mathcal{K}$.

The multiplicative nature of the guarantee implies that an output whose probability is zero on a given database must also have probability zero on any neighboring database, and hence, by repeated application of the definition, on any other database. Thus, Definition 1 trivially rules out the subsample-and-release paradigm discussed: For an individual x not in the dataset, the probability that x's data is sampled and released is obviously zero; the multiplicative nature of the guarantee ensures that the same is true for an individual whose data *is* in the dataset.

Any mechanism satisfying this definition addresses all concerns that any participant might have about the leakage of his or her personal information, regardless of any auxiliary information known to an adversary: Even if the participant removed his or her data from the dataset, no outputs (and thus consequences of outputs) would become significantly more or less likely. For example, if the database were to be consulted by an insurance provider before deciding whether or not to insure a given individual, then the presence or absence of *any* individual's data in the database will not significantly affect his or her chance of receiving coverage.

Definition 1 extends naturally to group privacy. Repeated application of the definition bounds the ratios of probabilities of outputs when a collection C of participants opts in or opts out, by a factor of $e^{|C|\varepsilon}$. Of course, the point of the statistical database is to disclose aggregate information about large groups (while simultaneously protecting individuals), so we should expect privacy bounds to disintegrate with increasing group size.

The parameter ε is public, and its selection is a social question. We tend to think of ε as, say, 0.01, 0.1, or in some cases, ln 2 or ln 3.

Sometimes, for example, in the census, an individual's participation is known, so hiding presence or absence makes no sense; instead we wish to hide the values in an individual's row. Thus, we can (and sometimes do) extend "differing in at most one row" to mean having symmetric difference at most 1 to capture both possibilities. However, we will continue to use the original definition.

Returning to randomized response, we see that it yields ε-differential privacy for a value of ε that depends on the universe from which the rows are chosen and the probability with which a random, rather than non-random, value is contributed by the respondent. As an example, suppose each row consists of a single bit, and that the respondent's instructions are to first flip an unbiased coin to determine whether he or she will answer randomly or truthfully. If heads (respond randomly), then the respondent is to flip a second unbiased coin and report the outcome; if tails, the respondent answers truthfully. Fix $b \in \{0, 1\}$. If the true value of the input is b, then b is output with probability 3/4. On the other hand, if the true value of the input is $1 - b$, then b is output with probability 1/4. The ratio is 3, yielding (ln 3)-differential privacy.

Suppose n respondents each employ randomized response independently, but using coins of known, fixed, bias. Then, given the randomized data, by the properties of the binomial distribution the analyst can approximate the true answer to the question "How many respondents have value b?" to within an expected error on the order of $\Theta(\sqrt{n})$. As we will see, it is possible to do much better—obtaining *constant* expected error, independent of n.

Generalizing in a different direction, suppose each row now has two bits, each one randomized independently, as described earlier. While each bit remains (ln 3)-differentially private, their logical-AND enjoys less privacy. That is, consider a privacy mechanism in which each bit is protected by this exact method of randomized response, and consider the query: "What is the logical-AND of the bits in the row of respondent i (after randomization)?" If we consider the two extremes, one in which respondent i has data 11 and the other in which respondent i has data 00, we see that in the first case the probability of output 1 is 9/16, while in the second case the probability is 1/16. Thus, this mechanism is at best (ln 9)-differentially private, not ln 3. Again, it is possible to do much better, even while releasing the entire 4-element histogram, also known as a *contingency table*, with only constant expected error in each cell.

Achieving Differential Privacy

Achieving differential privacy revolves around hiding the presence or absence of a single individual. Consider the query "How many rows in the database satisfy property P?" The presence or absence of a single row can affect the answer by at most 1. Thus, a differentially private mechanism for a query of this type can be designed by first computing the true answer and then adding random noise according to a distribution with the following property:

$$\forall z, z' \; s.t. \; |z - z'| = 1 : \Pr[z] \leq e^{\varepsilon} \Pr[z']. \quad (2)$$

To see why this is desirable, consider any feasible response r. For any m, if m is the true answer and the response is r then the random noise must have value $r - m$; similarly, if $m - 1$ is the true answer and the response is r, then the random noise must have value $r - m + 1$. In order for the response r to be generated in a differentially private fashion, it suffices for

$$e^{-\varepsilon} \leq \frac{\Pr[\text{noise} = r - m]}{\Pr[\text{noise} = r - m + 1]} \leq e^{\varepsilon}.$$

In general we are interested in vector-valued queries; for example, the data may be points in $\mathbf{R}^d$ and we wish to carry out an analysis that clusters the points and reports the location of the largest cluster.

Definition 2. *For* $f : \mathcal{D} \rightarrow \mathbf{R}^d$, *the* L_1 *sensitivity of* f *is*[7]

$$\Delta f = \max_{D,D'} \| f(D) - f(D') \|_1$$
$$= \max_{D,D'} \sum_{i=1}^{d} | f(D)_i - f(D')_i | \quad (3)$$

for all D, D' *differing in at most one row.*

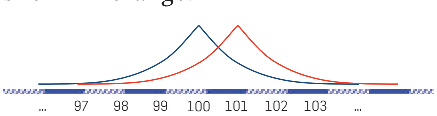
In particular, when $d = 1$ the sensitivity of f is the maximum difference in the values that the function f may take on a pair of databases that differ in only one row. This is the difference our noise must be designed to hide. For now, let us focus on the case $d = 1$.

The Laplace distribution with parameter b, denoted Lap(b), has density function $P(z|b) = \frac{1}{2b}\exp(-|z|/b)$; its variance is $2b^2$. Taking $b = 1/\varepsilon$ we have that the density at z is proportional to $e^{-\varepsilon|z|}$. This distribution has highest density at 0 (good for accuracy), and for any z, z' such that $|z - z'| \leq 1$ the density at z is at most e^{ε} times the density at z', satisfying the condition in Equation 2. It is also symmetric about 0, and this is important. We cannot, for example, have a distribution that only yields non-negative noise. Otherwise the only databases on which a counting query could return a response of 0 would be databases in which no row satisfies the query. Letting D be such a database, and letting $D' = D \cup \{r\}$ for some row r satisfying the query, the pair D, D' would violate ε-differential privacy. Finally, the distribution gets flatter as ε decreases. This is correct: smaller ε means better privacy, so the noise density should be less "peaked" at 0 and change more gradually as the magnitude of the noise increases.

There is nothing special about the cases $d = 1$, $\Delta f = 1$:

Theorem 2. *For* $f : \mathcal{D} \rightarrow \mathbf{R}^d$, *the mechanism* $\mathcal{K}$ *that adds independently generated noise with distribution Lap* $(\Delta f/\varepsilon)$ *to each of the d output terms enjoys* ε*-differential privacy.*[7]

Before proving the theorem, we illustrate the situation for the case of a counting query ($\Delta f = 1$) when $\varepsilon = \ln 2$ and the true answer to the query is 100. The distribution on the outputs (in gray) is centered at 100. The distribution on outputs when the true answer is 101 is shown in orange.



Proof. (Theorem 2) The proof is a simple generalization of the reasoning we used to illustrate the case of a single counting query.

Consider any subset $S \subseteq \text{Range}(\mathcal{K})$, and let D, D' be any pair of databases differing in at most one row. When the database is D, the probability density at any $r \in S$ is proportional to $\exp(-\| f(D) - r\|_1(\varepsilon/\Delta f))$. Similarly, when the database is D', the probability density at any $r \in \text{Range}(K)$ is proportional to $\exp(-\| f(D') - r\|_1(\varepsilon/\Delta f))$.

We have

$$\frac{e^{-\|f(D)-r\|(\varepsilon/\Delta f)}}{e^{-\|f(D')-r\|(\varepsilon/\Delta f)}} = \frac{e^{\|f(D')-r\|(\varepsilon/\Delta f)}}{e^{\|f(D)-r\|(\varepsilon/\Delta f)}}$$
$$= e^{(\|f(D')-r\| - \|f(D)-r\|)(\varepsilon/\Delta f)}$$
$$\leq e^{(\|f(D')-f(D)\|)(\varepsilon/\Delta f)}$$

where the inequality follows from the triangle inequality. By definition of sensitivity, $\| f(D') - f(D)\|_1 \leq \Delta f$, and so the ratio is bounded by $\exp(\varepsilon)$. Integrating over S yields ε-differential privacy. □

Given any query sequence $f_1, \ldots, f_m$, ε-differential privacy can be achieved by running $\mathcal{K}$ with noise distribution $\text{Lap}\left(\sum_{i=1}^{m} \Delta f_i / \varepsilon\right)$ on *each* query, even if the queries are chosen adaptively, with each successive query depending on the answers to the previous queries. In other words, by allowing the quality of each answer to deteriorate in a controlled way with the sum of the sensitivities of the queries, we can maintain ε-differential privacy.

With this in mind, let us return to some of the suggestions we considered earlier. Recall that using the specific randomized response strategy described above, for a single Boolean attribute, yielded error $\Theta\left(\sqrt{n}\right)$ on databases of size n and ($\ln 3$)-differential privacy. In contrast, using Theorem 2 with the same value of ε, noting that $\Delta f = 1$ yields a variance of $2(1/\ln 3)^2$, or an expected error of $\sqrt{2}/\ln 3$. More generally, to obtain ε-differential privacy we get an expected error of $\sqrt{2}/\varepsilon$. Thus, our expected error magnitude is constant, independent of n.

What about two queries? The sensitivity of a sequence of two counting queries is 2. Applying the theorem with $\Delta f/\varepsilon = 2/\varepsilon$, adding independently generated noise distributed as $\text{Lap}(2/\varepsilon)$ to each true answer yields ε-differential privacy. The variance is $2(2/\varepsilon)^2$, or standard deviation $2\sqrt{2}/\varepsilon$. Thus, for any desired ε we can achieve ε-differential privacy by increasing the expected magnitude of the errors as a function of the total sensitivity of the two-query sequence. This holds equally for:

- Two instances of the *same query*, addressing the repeated query problem
- One count for each of two different bit positions, for example, when each row consists of two bits
- A pair of queries of the form: "How many rows satisfy property P?" and "How many rows satisfy property Q?" (where possibly $P = Q$)
- An arbitrary pair of queries

However, the theorem also shows we can sometimes do better. The logical-AND count we discussed earlier, even though it involves two different bits in each row, still only has sensitivity 1: The number of 2-bit rows whose entries are both 1 can change by at most 1 with the addition or deletion of a single row. Thus, this more complicated query can be answered in an ε-differentially private fashion using noise distributed as Lap($1/\varepsilon$); we do not need to use the distribution Lap($2/\varepsilon$).

Histogram Queries. The power of Theorem 2 really becomes clear when considering *histogram queries*, defined as follows. If we think of the rows of the database as elements in a universe X, then a histogram query is a partitioning of X into an arbitrary number of disjoint regions $X_1, X_2, \ldots, X_d$. The implicit question posed by the query is: "For $i = 1, 2, \ldots, d$, how many points in the database are contained in X_i?" For example, the database may contain the annual income for each respondent, and the query is a partitioning of incomes into ranges: $\{[0, 50K), [50K, 100K), \ldots, \geq 500K\}$. In this case $d = 11$, and the question is asking, for each of the d ranges, how many respondents in the database have annual income in the given range. This looks like d separate counting queries, but the entire query actually has sensitivity $\Delta f = 1$. To see this, note that if we remove one row from the database, then only one cell in the histogram changes, and that cell only changes by 1; similarly for adding a single row. So Theorem 2 says that ε-differential privacy can be maintained by perturbing each cell with an independent random draw from Lap($1/\varepsilon$). Returning to our example of 2-bit rows, we can pose the 4-ary histogram query requesting, for each pair of literals v_1v_2, the number of rows with value v_1v_2, adding noise of order $1/\varepsilon$ to each of the four cells.

When Noise Makes No Sense. There are times when the addition of noise for achieving privacy makes no sense. For example, the function f might map databases to strings, strategies, or trees, or it might be choosing the "best" among some specific, not necessarily continuous, set of real-valued objects. The problem of optimizing the output of such a function while preserving ε-differential privacy requires additional technology.

There are times when the addition of noise for achieving privacy makes no sense.

Assume the curator holds a database D and the goal is to produce an object y. The *exponential mechanism*[19] works as follows. We assume the existence of a *utility function* $u(D, y)$ that measures the quality of an output y, given that the database is D. For example, the data may be a set of labeled points in $\mathbf{R}^d$ and the output y might be a d-ary vector describing a $(d - 1)$-dimensional hyperplane that attempts to classify the points, so that those labeled with +1 have non-negative inner product with y and those labeled with –1 have negative inner product. In this case the utility would be the number of points correctly classified, so that higher utility corresponds to a better classifier. The exponential mechanism outputs y with probability proportional to $\exp(u(D,y)\varepsilon/\Delta u)$ and ensures ε-differential privacy. Here Δu is the sensitivity of the utility function bounding, for all databases (D, D') differing in only one row and potential outputs y, the difference $|u(D, y) - u(D', y)|$. In our example, $\Delta u = 1$. The mechanism assigns most mass to the best classifier, and the mass assigned to any other drops off exponentially in the decline in its utility for the current dataset—hence the name "exponential mechanism."

When Sensitivity Is Hard to Analyze. The Laplace and exponential mechanisms provide a differentially private interface through which the analyst can access the data. Such an interface can be useful even when it is difficult to determine the sensitivity of the desired function or query sequence; it can also be used to run an iterative algorithm, composed of easily analyzed steps, for as many iterations as a given privacy budget permits. This is a powerful observation; for example, using only noisy sum queries, it is possible to carry out many standard data mining tasks, such as singular value decompositions, finding an ID3 decision tree, clustering, learning association rules, and learning anything learnable in the statistical queries learning model, frequently with good accuracy, in a privacy-preserving fashion.[2] This approach has been generalized to yield a publicly available codebase for writing programs that ensure differential privacy.[18]

***k*-Means Clustering.** As an example of "private programming,"[2] consider k-means clustering, described first in its usual, non-private form. The input consists of points $p_1, \ldots, p_n$ in the d-dimensional unit cube $[0, 1]^d$. Initial candidate means $\mu_1, \ldots, \mu_k$ are chosen randomly from the cube and updated as follows:

1. Partition the samples $\{p_i\}$ into k sets $S_1, \ldots, S_k$, associating each p_i with the nearest μ_j.
2. For $1 \le j \le k$, set $\mu'_j = \sum_{i \in S_j} p_i / |S_j|$, the mean of the samples associated with μ_j.

This update rule is typically iterated until some convergence criterion has been reached, or a fixed number of iterations have been applied.

Although computing the nearest mean of any one sample (Step 1) would breach privacy, we observe that to compute an average among an unknown set of points it is enough to compute their sum and divide by their number. Thus, the computation only needs to expose the approximate cardinalities of the S_j, not the sets themselves. Happily, the k candidate means implicitly define a histogram query, since they partition the space $[0, 1]^d$ according to their Voronoi cells, and so the vector $(|S_1|, \ldots, |S_k|)$ can be released with very low noise in each coordinate. This gives us a differentially private approximation to the denominators in Step 2. As for the numerators, the sum of a subset of the p_i has sensitivity at most d, since the points come from the bounded region $[0,1]^d$. Even better, the sensitivity of the d-ary function that returns, for each of the k Voronoi cells, the d-ary sum of the points in the cell is at most d: Adding or deleting a single d-ary point can affect at most one sum, and that sum can change by at most 1 in each of the d dimensions. Thus, using a query sequence with total sensitivity at most $d + 1$, the analyst can compute a new set of candidate means by dividing, for each μ_j, the approximate sum of the points in S_j by the approximation to the cardinality $|S_j|$.

If we run the algorithm for a fixed number N of iterations we can use the noise distribution $\mathrm{Lap}((d + 1)N/\varepsilon)$ to obtain ε-differential privacy. If we do not know the number of iterations in advance we can increase the noise parameter as the computation proceeds. There are many ways to do this. For example, we can answer in the first iteration with parameter $(d + 1)(\varepsilon/2)$, in the next with parameter $(d + 1)(\varepsilon/4)$, and so on, each time using up half of the remaining "privacy budget."

Generating Synthetic Data

The idea of creating a synthetic dataset whose statistics closely mirror those of the original dataset, but which preserves privacy of individuals, was proposed in the statistics community no later than 1993.[21] The lower bounds on noise discussed at the end of Section on "How Is Hard" imply that no such dataset can safely provide very accurate answers to too many weighted subset sum questions, motivating the interactive approach to private data analysis discussed herein. Intuitively, the advantage of the interactive approach is that only the questions actually asked receive responses.

Against this backdrop, the non-interactive case was revisited from a learning theory perspective, challenging the interpretation of the noise lower bounds as a limit on the number of queries that can be answered privately.[3] This work, described next, has excited interest in interactive and non-interactive solutions yielding noise in the range $\left[\omega\left(\sqrt{n}\right), o(n)\right]$.

Let X be a universe of data items and let $\mathcal{C}$ be a *concept class* consisting of functions $c : X \to \{0,1\}$. We say $x \in X$ *satisfies* a concept $c \in \mathcal{C}$ if and only if $c(x) = 1$. A concept class can be extremely general; for example, it might consist of all rectangles in the plane, or all Boolean circuits containing a given number of gates.

Given a sufficiently large database $D \in X^n$, it is possible to privately generate a synthetic database that maintains approximately correct fractional counts for *all* concepts in $\mathcal{C}$ (there may be infinitely many!). That is, letting S denote the synthetic database produced, with high probability over the choices made by the privacy mechanism, for every concept $c \in \mathcal{C}$, the fraction of elements in S that satisfy c is approximately the same as the fraction of elements in D that satisfy c.

The minimal size of the input database depends on the quality of the approximation, the logarithm of the cardinality of the universe X, the privacy parameter ε, and the *Vapnick–Chervonenkis dimension* of the concept class $\mathcal{C}$ (for finite $|C|$ this is at most $\log_2 |\mathcal{C}|$). The synthetic dataset, chosen by the exponential mechanism, will be a set of $m = O(\mathrm{VCdim}(\mathcal{C})/\gamma^2)$, elements in X (γ governs the maximum permissible inaccuracy in the fractional count). Letting D denote the input dataset and $\hat{D}$ a candidate synthetic dataset, the utility function for the exponential mechanism is given by

$$u(D, \hat{D}) = -\max_{h \in \mathcal{C}} \left| h(D) - \frac{n}{m} h(\hat{D}) \right|.$$

Pan-Privacy

Data collected by a curator for a given purpose may be subject to "mission creep" and legal compulsion, such as a subpoena. Of course, we could analyze data and then throw it away, but can we do something even stronger, never storing the data in the first place? Can we strengthen our notion of privacy to capture the "never store" requirement?

These questions suggest an investigation of differentially private streaming algorithms with small state—much too small to store the data. However, nothing in the definition of a streaming algorithm, even one with very small state, precludes storing a few individual data points. Indeed, popular techniques from the streaming literature, such as Count-Min Sketch and subsampling, do precisely this. In such a situation, a subpoena or other intrusion into the local state will breach privacy.

A *pan-private* algorithm is private "inside and out," remaining differentially private even if its internal state becomes visible to an adversary.[10] To understand the pan-privacy guarantee, consider click stream data. This data is generated by individuals, and an individual may appear many times in the stream. Pan-privacy requires that any two streams differing only in the information of a single individual should produce very similar distributions on the *internal states* of the algorithm *and on its outputs*, even though the data of an individual are interleaved arbitrarily with other data in the stream.

As an example, consider the problem of *density estimation*. Assuming, for simplicity, that the data stream is just a sequence of IP addresses in a certain range, we wish to know what fraction of the set of IP addresses in the range actually appears in the stream. A solution inspired by randomized response can be designed using the following technique.[10]

Define two probability distributions, D_0 and D_1, on the set $\{0, 1\}$. D_0 assigns equal mass to zero and to one. D_1 has a slight bias toward 1; specifically, 1 has mass $1/2 + \varepsilon/4$, while 0 has mass $1/2 - \varepsilon/4$.

Let X denote the set of all possible IP addresses in the range of interest. The algorithm creates a table, with a 1-bit entry b_x for each $x \in X$, initialized to an independent random draw from D_0. So initially the table is roughly half zeroes and half ones.

In an atomic step, the algorithm receives an element from the stream, changes state, and discards the element. When processing $x \in X$, the algorithm makes a fresh random draw from D_1, and stores the result in b_x. This is done no matter how many times x may have appeared in the past. Thus, for any x appearing at least once, b_x will be distributed according to D_1. However, if x never appears, then the entry for x is the bit drawn according to D_0 during the initialization of the table.

As with randomized response, the density in X of the items in the stream can be approximated from the number of 1's in the table, taking into account the expected fraction of "false positives" from the initialization phase and the "false negatives" when sampling from D_1. Letting θ denote the fraction of entries in the table with value 1, the output is $4(\theta - 1/2)/\varepsilon + \mathrm{Lap}(1/\varepsilon|\boldsymbol{X}|)$.

Intuitively, the internal state is differentially private because, for each $b \in \{0, 1\}$, $e^{-\varepsilon} \le \Pr_{D_1}[b]/\Pr_{D_0}[b] \le e^{\varepsilon}$; privacy for the output is ensured by the addition of Laplacian noise. Over all, the algorithm is 2ε-differentially pan-private.

Conclusion

The differential privacy frontier is expanding rapidly, and there is insufficient space here to list all the interesting directions currently under investigation by the community. We identify a few of these.

The Geometry of Differential Privacy. Sharper upper and lower bounds on noise required for achieving differential privacy against a sequence of linear queries can be obtained by understanding the geometry of the query sequence.[14] In some cases dependencies among the queries can be exploited by the curator to markedly improve the accuracy of the responses. Generalizing this investigation to the nonlinear and interactive cases would be of significant interest.

Algorithmic Complexity. We have so far ignored questions of computational complexity. Many, but not all, of the techniques described here have efficient implementations. For example, there are instances of the synthetic data generation problem that, under standard cryptographic assumptions, have no polynomial time implementation.[11] It follows that there are cases in which the exponential mechanism has no efficient implementation. When can this powerful tool be implemented efficiently, and how?

An Alternative to Differential Privacy? Is there an alternative, "*ad omnia*," guarantee that composes automatically, and permits even better accuracy than differential privacy? Can cryptography be helpful in this regard?[20]

The work described herein has, for the first time, placed private data analysis on a strong mathematical foundation. The literature connects differential privacy to decision theory, economics, robust statistics, geometry, additive combinatorics, cryptography, complexity theory learning theory, and machine learning. Differential privacy thrives because it is natural, it is not domain-specific, and it enjoys fruitful interplay with other fields. This flexibility gives hope for a principled approach to privacy in cases, like private data analysis, where traditional notions of cryptographic security are inappropriate or impracticable. C

References

1. Adam, N.R., Wortmann, J. Security-control methods for statistical databases: A comparative study. *ACM Comput. Surv. 21* (1989), 515–556.
2. Blum, A., Dwork, C., McSherry, F., Nissim, K. Practical privacy: The SuLQ framework. In *Proceedings of the 24th ACM Symposium on Principles of Database Systems* (2005), 128–138.
3. Blum, A., Ligett, K., Roth, A. A learning theory approach to non-interactive database privacy. In *Proceedings of the 40th ACM Symposium on Theory of Computing* (2008), 609–618.
4. Denning, D.E. Secure statistical databases with random sample queries. *ACM Trans. Database Syst. 5* (1980), 291–315.
5. Dinur, I., Nissim, K. Revealing information while preserving privacy. In *Proceedings of the 22nd ACM Symposium on Principles of Database Systems* (2003), 202–210.
6. Dwork, C. Differential privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP) (2)* (2006), 1–12.
7. Dwork, C., McSherry, F., Nissim, K., Smith, A. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Theory of Cryptography Conference* (2006), 265–284.
8. Dwork, C., McSherry, F., Talwar, K. The price of privacy and the limits of lp decoding. In *Proceedings of the 39th ACM Symposium on Theory of Computing* (2007), 85–94.
9. Dwork, C. Naor, M. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *J. Privacy Confidentiality 2* (2010). Available at: http://repository.cmu.edu/jpc/vol2/iss1/8.
10. Dwork, C., Naor, M., Pitassi, T., Rothblum, G., Yekhanin, S. Pan-private streaming algorithms. In *Proceedings of the 1st Symposium on Innovations in Computer Science* (2010).
11. Dwork, C., Naor, M., Reingold, O., Rothblum, G., Vadhan, S. When and how can privacy-preserving data release be done efficiently? In *Proceedings of the 41st ACM Symposium on Theory of Computing* (2009), 381–390.
12. Dwork, C., Nissim, K. Privacy-preserving datamining on vertically partitioned databases. In *Advances in Cryptology—CRYPTO'04* (2004), 528–544.
13. Goldwasser, S., Micali, S. Probabilistic encryption. *JCSS 28* (1984), 270–299.
14. Hardt, M., Talwar, K. On the geometry of differential privacy, (2009). In *Proceedings of the 42nd ACM Symposium on Theory of Computing* (2010), 705–714.
15. Kenthapadi K., Mishra, N., Nissim, K. Simulatable auditing. In *Proceedings of the 24th ACM Symposium on Principles of Database Systems* (2005), 118–127.
16. Kleinberg, J., Papadimitriou, C., Raghavan, P. Auditing boolean attributes. In *Proceedings of the 19th ACM Symposium on Principles of Database Systems* (2000), 86–91.
17. Kumar, R., Novak, J., Pang, B., Tomkins, A. On anonymizing query logs via token-based hashing. In *Proceedings of the WWW 2007* (2007), 629–638.
18. McSherry, F. Privacy integrated queries (codebase). Available on Microsoft Research downloads website. See also *Proceedings of SIGMOD* (2009), 19–30.
19. McSherry, F., Talwar, K. Mechanism design via differential privacy. In *Proceedings of the 48th Annual Symposium on Foundations of Computer Science* (2007).
20. Mironov, I., Pandey, O., Reingold, O., Vadhan, S. Computational differential privacy. In *Advances in Cryptology—CRYPTO'09* (2009), 126–142.
21. Rubin, D. Discussion: Statistical disclosure limitation. *J. Official Statist. 9* (1993), 462–468.
22. Sweeney, L. Weaving technology and policy together to maintain confidentiality. *J. Law Med. Ethics 25* (1997), 98–110.
23. Warner, S. Randomized response: a survey technique for eliminating evasive answer bias. *JASA* (1965), 63–69.

Cynthia Dwork (dwork@microsoft.com) is a principal researcher at Microsoft Research, Silicon Valley Campus, Mountain View, CA.

[CONTINUED FROM P. 16] is not solely about communication; it is also about maintaining a historical record, so that future generations of scientists can learn and build on the work of the past. Whether new forms of scholarly communication pass this second test is far from certain.

A common misconception about the "dead tree" model of scholarly communication is that it is antithetical to speed. This is only true to a certain extent, but almost certainly not to the extent that most believe.

Scott Delman, ACM's Director of Group Publishing, commented that "The current system of peer review is the single largest bottleneck in the scholarly publication process, but this does not mean the established system can simply be thrown out in favor of new models just because new technology enables dramatic improvements in speed." Establishing a new model for scholarly communication will involve experimentation, trial and error, and most likely evolution instead of revolution. Proclamations of the death of scholarly publishers and scholarly publishing as a result of the rise of the Internet are no longer taken seriously by those working in the publishing industry. What we have seen is a slow but steady evolution of print to online publication and distribution models instead of an overnight upheaval.

Delman adds, "I believe strongly that there is a need for a new model," but then goes on to refute the notion that digital-only publishing—and the elimination of print—would quicken the publication of scholarly articles. "The most substantial component in the time delay related to the publication of articles in scholarly journals is the peer-review process," and a digital-only model won't change that, he says. Nor will it reduce article backlogs or remove page limitations. "Eliminating print will not have the dramatic impact that most assume will occur if print publications go away," he says. Importantly, ACM readers and subscribers "look for high-quality content delivered in multiple formats, and they still want print."

Adding to the complexity of the challenge is the fact that while science is global, scientific publication models are often socially or geographically influenced, so there is no single solution that can be identified to improve the speed or efficiency of scholarly communication. Ed Chi, of the Palo Alto Research Center, described some of the difficulties of modernizing peer-review publishing in the Blog@CACM at (http://cacm.acm.org/blogs/blog-cacm/100284). "In many non-U.S. research evaluations, only the ISI Science Citation Index actually counts for publication. Already this doesn't fit with many real-world metrics for reputation," Chi said via email. Some well-known ACM conference publications are excluded from the SCI (http://bit.ly/iaobEa), "even when their real-world reputation is much higher than other 'dead-tree' journals." Technology may provide opportunities to facilitate and accelerate the discourse, but there is no guarantee the academic establishment around the world will move as quickly in accepting new media and ways of communicating.

Paperless publishing will happen gradually, but "only if there are ways to manage the publication process," Chi says. "Open source journal publication management systems will enable journals to go somewhat independent of traditional paper publishers, but we will also need national scientific institutions to establish digital archives." Other challenges he notes include handling an increased number of submissions and managing potentially larger editorial boards.

As an organization with the stated mission to advance computing as a science and profession, ACM could "lead the charge" in experimenting with new digital publishing models for computing scholarship, says Chi. "This might include creating usable software, digital libraries, or archival standards." A particularly important area of research would examine how to make socially derived metrics a part of reputation systems, so that the number of downloads, online mentions, citations, and blog discussions can be measured for influence. Then, according to Chi, ACM "should work with national libraries to actively change the publication models of other professions and fields." This will not be a revolution. ACM can help to drive the change in a positive way for the scientific community.

DOI:10.1145/1866739.1866759

Technical Perspective
Sora Promises Lasting Impact

By Dina Katabi

THE TERM SOFTWARE defined radio (SDR) first appeared in 1992 and referred to a radio transceiver where the basic signal processing components (for example, filtering, frame detection, synchronization, and demodulation) are all done in a general-purpose processor. The goal of an SDR was to enable a single radio to support multiple wireless technologies (for example, AM, VHF, FM) and be easily upgradable with a software patch.

While the concept of SDR has been around for decades, only recently have SDRs become common in academic wireless research. However, research projects typically employ SDR as a development platform, that is, they use software radios to develop new physical layer designs with the understanding that if these designs make it to a product they will be built in ASICs. The reason why SDR has become a development platform rather than a fully functional software radio is that building high-performance SDRs has turned out to be very challenging.

Sora has revived the original SDR vision. The objective of Sora is to build an SDR that combines the performance and fidelity of hardware platforms with the programmability and flexibility of general-purpose processors. To do so, Sora must overcome the following challenge: How can a radio deliver high throughput and support real-time protocols when all signal processing is done in software on a PC?

There are many reasons why the following paper about Sora stands out as one of the most significant wireless papers in the past few years.

Sora's approach uses various features common in today's multicore architectures. For example, transferring the digital waveform samples from the radio board to the PC requires very high bus throughput. While alternative SDR technologies employ USB 2.0 or Gigabit Ethernet, Sora opts for PCI-Express. This design decision enables Sora to achieve significantly higher transfer rates, which are important for high bandwidth multi-antenna designs. The choice of PCI-express also enables Sora to reduce the transfer latency to sub-microseconds, which is necessary for wireless protocols with timing constraints (for example, MAC protocols). Further, to accelerate wireless processing, Sora replaces computation with memory lookups, exploits single instruction multiple data (SIMD), and dedicates certain cores exclusively to real-time signal processing.

There are many reasons why the following paper about Sora stands out as one of the most significant wireless papers in the past few years. First, it presents the first SDR platform that fully implements IEEE 802.11b/g on standard PCs. Second, the design choices it makes (for example, the use of PCIe, SIMD, trading computation for memory lookups, and core dedication) are highly important if software radios are ever to meet their original goal of one-radio-for-all-wireless-technologies. Third, the paper is a beautiful and impressive piece of engineering that spans signal processing, hardware design, multicore programming, kernel optimization, and so on. For all these reasons, this paper will have a lasting impact on wireless research.

The Sora platform has been used in multiple research projects and real-time demos. It has enabled demanding designs, such as LTE and AP virtualization, to be built fully in software. However, currently most SDR-based research uses the GNU Radio/USRP platform. Despite the limitations of this platform, previous attempts at replacing it with more capable platforms did not experience significant success. In fact, history shows that wide adoption is not necessarily correlated with the more capable design. One of the classic papers we teach our undergraduate students is "The Rise of Worse is Better" by Richard Gabriel that explains why the Lisp language lost to C and Unix. Gabriel argues that for wide adoption, a system must be good enough and as simple as possible. Such a design (termed worse is better) tends to appear first because the implementer did not spend an excessive amount of time over-optimizing. Therefore, if good enough, it will be adopted by developers because of its simplicity. Once adopted, the system will gradually improve until it is *almost* the right design. One may argue that the history of the GNU Radio/USRP SDR is fairly similar; the platform originally provided just enough for people to start experimenting with the wireless physical layer. As a result, it was simple and cheap, which caused it to spread. Once it was accepted, it kept improving just enough to enable the next step in research.

The Sora team has recently started a program that awards Sora kits to academic institutions to enable them to experiment with this new platform. It will be interesting to see whether Sora with its higher performance can eventually replace the GNU Radio/USRP platform. If this happens, it will be a major success for Sora.

Dina Katabi (dina@csail.mit.edu) is an associate professor in the Electrical Engineering and Computer Science Department at Massachusetts Institute of Technology, Cambridge, MA.

DOI:10.1145/1866739.1866760

Sora: High-Performance Software Radio Using General-Purpose Multi-Core Processors

By Kun Tan, He Liu, Jiansong Zhang, Yongguang Zhang, Ji Fang, and Geoffrey M. Voelker

Abstract

This paper presents Sora, a fully programmable software radio platform on commodity PC architectures. Sora combines the performance and fidelity of hardware software-defined radio (SDR) platforms with the programmability and flexibility of general-purpose processor (GPP) SDR platforms. Sora uses both hardware and software techniques to address the challenges of using PC architectures for high-speed SDR. The Sora hardware components consist of a radio front-end for reception and transmission, and a radio control board for high-throughput, low-latency data transfer between radio and host memories. Sora makes extensive use of features of contemporary processor architectures to accelerate wireless protocol processing and satisfy protocol timing requirements, including using dedicated CPU cores, large low-latency caches to store lookup tables, and SIMD processor extensions for highly efficient physical layer processing on GPPs. Using the Sora platform, we have developed a few demonstration wireless systems, including SoftWiFi, an 802.11a/b/g implementation that seamlessly interoperates with commercial 802.11 NICs at all modulation rates, and SoftLTE, a 3GPP LTE uplink PHY implementation that supports up to 43.8Mbps data rate.

1. INTRODUCTION

Software-defined radio (SDR) holds the promise of fully programmable wireless communication systems, effectively supplanting current technologies which have the lowest communication layers implemented primarily in fixed, custom hardware circuits. Realizing the promise of SDR in practice, however, has presented developers with a dilemma.

Many current SDR platforms are based on either programmable hardware such as field programmable gate arrays (FPGAs)[8, 10] or embedded digital signal processors (DSPs).[6, 12] Such hardware platforms can meet the processing and timing requirements of modern high-speed wireless protocols, but programming FPGAs and specialized DSPs are difficult tasks. Developers have to learn how to program to each particular embedded architecture, often without the support of a rich development environment of programming and debugging tools. Such hardware platforms can also be expensive.

In contrast, SDR platforms based on general-purpose processor (GPP) architectures, such as commodity PCs, have the opposite set of trade-offs. Developers program to a familiar architecture and environment using sophisticated tools, and radio front-end boards for interfacing with a PC are relatively inexpensive. However, since PC hardware and software have not been designed for wireless signal processing, existing GPP-based SDR platforms can achieve only limited performance.[1, 7] For example, the popular USRP/GNU Radio platform is reported to achieve only 100kbps throughput on an 8-MHz channel,[18] whereas modern high-speed wireless protocols like 802.11 support multiple Mbps data rates on a much wider 20-MHz channel. These constraints prevent developers from using such platforms to achieve the full fidelity of state-of-the-art wireless protocols while using standard operating systems and applications in a real environment.

In this paper we present Sora, a fully programmable software radio platform that provides the benefits of both SDR approaches, thereby resolving the SDR platform dilemma for developers. With Sora, developers can implement and experiment with high-speed wireless protocol stacks, e.g., IEEE 802.11a/b/g and 3GPP LTE, using commodity general-purpose PCs. Developers program in familiar programming environments with powerful tools on standard operating systems. Software radios implemented on Sora appear like any other network device, and users can run unmodified applications on their software radios with the same performance as commodity hardware wireless devices.

An implementation of high-speed wireless protocols on general-purpose PC architectures must overcome a number of challenges that stem from existing hardware interfaces and software architectures. First, transferring high-fidelity digital waveform samples into PC memory for processing requires very high bus throughput. For example, existing 802.11a/b/g requires 1.2Gbps system throughput to transfer digital signals for a single 20-MHz channel, while the latest 802.11n standard needs near 10Gbps as it uses even wider band and multiple-input–multiple-output (MIMO) technology. Second, physical layer (PHY) signal processing requires high computation for generating information bits from the large amount of digital samples, and vice versa, particularly at high modulation rates; indeed, back-of-the-envelope calculations for processing requirements on GPPs have instead

The original version of this paper was published in *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation (NSDI'09)*. This work was performed when Ji Fang and He Liu were visiting students and Geoffrey M. Voelker was a visiting researcher at Microsoft Research Asia.

motivated specialized hardware approaches in the past.[14, 16] Lastly, wireless PHY and media access control (MAC) protocols have low-latency real-time deadlines that must be met for correct operation. For example, the 802.11 MAC protocol requires precise timing control and ACK response latency on the order of tens of microseconds. Existing software architectures on the PC cannot consistently meet this timing requirement.

Sora addresses these challenges with novel hardware and software designs. First, we have developed a new, inexpensive radio control board (RCB) with a radio front-end for transmission and reception. The RCB bridges an RF front-end with PC memory over the high-speed and low-latency PCIe bus. With this bus standard, the RCB can support 16.7Gbps (×8 mode) throughput with sub-microsecond latency, which together satisfies the throughput and timing requirements of modern wireless protocols while performing all digital signal processing on host CPU and memory.

Second, to meet PHY processing requirements, Sora makes full use of various features of widely adopted multi-core architectures in existing GPPs. The Sora software architecture explicitly supports streamlined processing that enables components of the signal processing pipeline to efficiently span multiple cores. Further, we change the conventional implementation of PHY components to extensively take advantage of lookup tables (LUTs), trading off computation for memory. These LUTs substantially reduce the computational requirements of PHY processing, while at the same time taking advantage of the large, low-latency caches on modern GPPs. Finally, Sora uses the Single Instruction Multiple Data (SIMD) extensions in existing processors to further accelerate PHY processing.

Lastly, to meet the real-time requirements of high-speed wireless protocols, Sora provides a new kernel service, *core dedication*, which allocates processor cores exclusively for real-time SDR tasks. We demonstrate that it is a simple yet crucial abstraction that guarantees the computational resources and precise timing control necessary for SDR on a multi-core GPP.

We have developed a few demonstration wireless systems based on the Sora platform, including: (1) SoftWiFi, an 802.11a/b/g implementation that supports a full suite of modulation rates (up to 54Mbps) and seamlessly interoperates with commercial 802.11 NICs, and (2) SoftLTE, a 3GPP LTE uplink PHY implementation that supports up to 43.8Mbps data rate.

The rest of the paper is organized as follows. Section 2 provides background on wireless communication systems. We then present the Sora architecture in Section 3, and we discuss our approach for addressing the challenges of building an SDR platform on a GPP system in Section 4. We then describe the implementation of the Sora platform in Section 5. Section 6 provides a quantitative evaluation of the radio systems based on Sora. Finally, Section 7 describes related work and Section 8 concludes.

2. BACKGROUND AND REQUIREMENTS

In this section, we briefly review the PHY and MAC components of typical wireless communication systems. Although different wireless technologies may have subtle differences among one another, they generally follow similar designs and share many common algorithms. In this section, we use the IEEE 802.11a/b/g standards to exemplify characteristics of wireless PHY and MAC components as well as the challenges of implementing them in software.

2.1. Wireless PHY

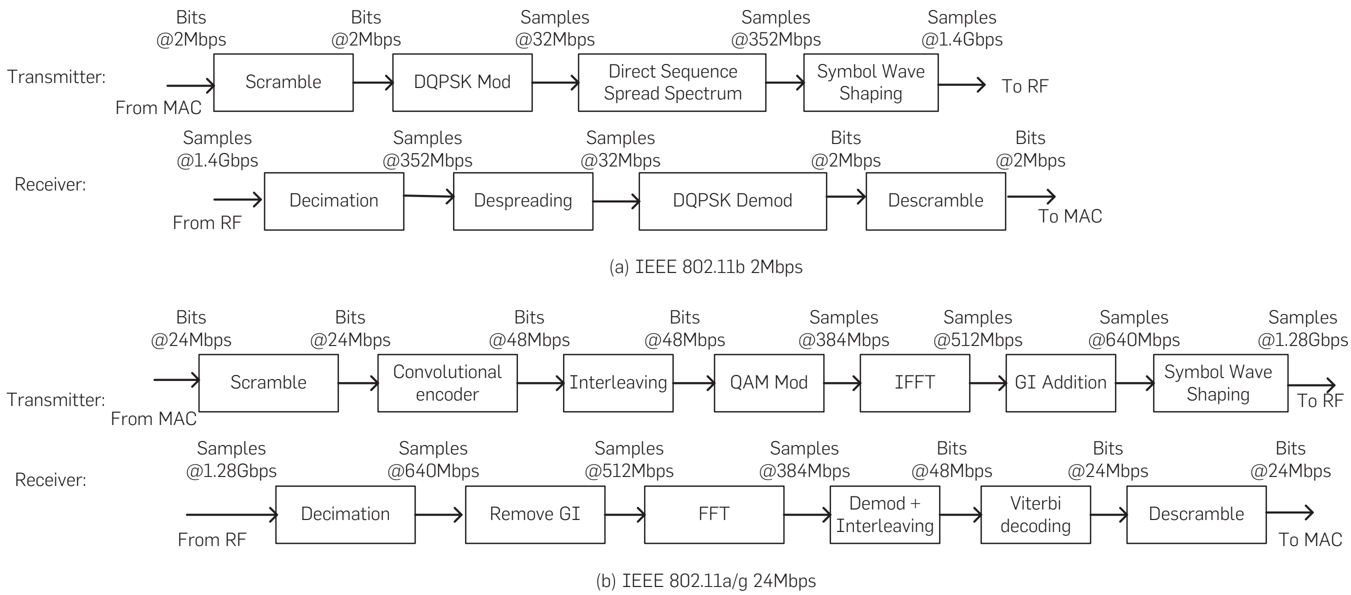
The role of the PHY layer is to convert information bits into a radio waveform, or vice versa. At the transmitter side, the wireless PHY component first *modulates* the message (i.e., a MAC frame) into a time sequence of *digital baseband signals*. Digital baseband signals are then passed to the radio front-end, where they are converted to analog waveform, multiplied by a high frequency *carrier* and transmitted into the wireless channel. At the receiver side, the radio front-end receives radio signals in the channel and extracts the baseband waveform by removing the high-frequency carrier. The extracted baseband waveform is digitalized and converted back into digital signals. Then, the digital baseband signals are fed into the receiver's PHY layer to be *demodulated* into the original message.

The PHY layer directly operates on the digital baseband signals after modulation on the transmitter side and before demodulation on the receiver side. Therefore, high-throughput interfaces are needed to connect the PHY layer and the radio front-end. The required throughput linearly scales with the bandwidth of the baseband signal as well as the number of antennas in a MIMO system. For example, the channel width is 20MHz in 802.11a. It requires a data rate of at least 20M complex samples per second to represent the waveform. These complex samples normally require 16-bit quantization for both in-phase and quadrature (I/Q) components to provide sufficient fidelity, translating into 32 bits per sample, or 640Mbps for the full 20 MHz channel. Over-sampling, a technique widely used for better performance,[11] doubles the requirement to 1.28Gbps. With a 4 × 4 MIMO and 40-MHz channel, as specified in 802.11n, it will again quadruple the requirement to 10Gbps to move data between the RF frond-end and PHY for one channel.

Advanced communication systems (e.g., IEEE 802.11a/b/g, as shown in Figure 1) contain multiple functional blocks in their PHY components. These functional blocks are pipelined with one another. Data are streamed through these blocks sequentially, but with different data types and sizes. As illustrated in Figure 1, different blocks may consume or produce different types of data in different rates arranged in small data blocks. For example, in 802.11b, the scrambler may consume and produce one bit, while DQPSK modulation maps each two-bit data block onto a complex symbol, whose real and image components represent I and Q, respectively.

Each PHY block performs a fixed amount of computation on every transmitted or received bit. When the data rate is high, e.g., 11Mbps for 802.11b and 54Mbps for 802.11a/g, PHY processing blocks consume a significant amount of computational power. Based on the model in Neel et al.,[16] we estimate that a direct implementation of 802.11b may require 10GOPS while 802.11a/g needs at least 40GOPs.

Figure 1. PHY operations of IEEE 802.11a/b/g transceiver.



These requirements are very demanding for software processing in GPPs.

2.2. Wireless MAC

The wireless channel is a resource shared by all transceivers operating on the same spectrum. As simultaneously transmitting neighbors may interfere with each other, various MAC protocols have been developed to coordinate their transmissions in wireless networks to avoid collisions.

Most modern MAC protocols, such as 802.11, require timely responses to critical events. For example, 802.11 adopts a carrier sense multiple access (CSMA) MAC protocol to coordinate transmissions. Transmitters are required to sense the channel before starting their transmission, and channel access is only allowed when no energy is sensed, i.e., the channel is free. The latency between *sense* and *access* should be as small as possible. Otherwise, the sensing result could be outdated and inaccurate. Another example is the link-layer retransmission mechanisms in wireless protocols, which may require an immediate acknowledgement (ACK) to be returned in a limited time window.

Commercial standards like IEEE 802.11 mandate a response latency within 16 μs, which is challenging to achieve in software on a general-purpose PC with a general-purpose OS.

2.3. Software radio requirements

Given the above discussion, we summarize the requirements for implementing a software radio system on a general PC platform:

High-system throughput. The interfaces between the radio front-end and PHY as well as between some PHY processing blocks must possess sufficiently high throughput to transfer high-fidelity digital waveforms.

Intensive computation. High-speed wireless protocols require substantial computational power for their PHY processing. Such computational requirements also increase proportionally with communication speed. Unfortunately, techniques used in conventional PHY hardware or embedded DSPs do not directly carry over to GPP architectures. Thus, we require new software techniques to accelerate high-speed signal processing on GPPs. With the advent of many-core GPP architectures, it is now reasonable to aggregate computational power of multiple CPU cores for signal processing. But, it is still challenging to build a software architecture to efficiently exploit the full capability of multiple cores.

Real-time enforcement. Wireless protocols have multiple *real-time deadlines* that need to be met. Consequently, not only is processing throughput a critical requirement, but the processing latency needs to meet response deadlines. Some MAC protocols also require precise timing control at the granularity of microseconds to ensure certain actions occur at exactly pre-scheduled time points. Meeting such real-time deadlines on a general PC architecture is a nontrivial challenge: time sharing operating systems may not respond to an event in a timely manner, and bus interfaces, such as Gigabit Ethernet, could introduce indefinite delays far more than a few microseconds. Therefore, meeting these real-time requirements requires new mechanisms on GPPs.

3. ARCHITECTURE

We have developed a high-performance software radio platform called Sora that addresses these challenges. It is based on a commodity general-purpose PC architecture. For flexibility and programmability, we push as much communication functionality as possible into software, while keeping hardware additions as simple and generic as possible. Figure 2 illustrates the overall system architecture.

Figure 2. Sora system architecture. All PHY and MAC execute in software on a commodity multi-core CPU.

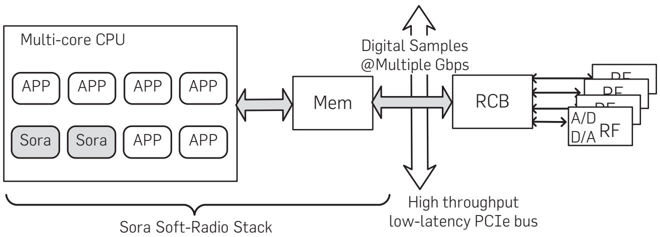
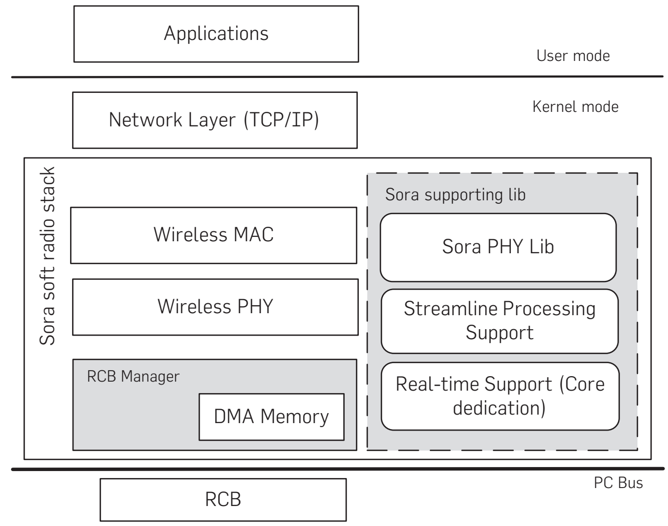


Figure 3. Software architecture of Sora soft-radio stack.



3.1. Hardware components

The hardware components in the Sora architecture are a new RCB with an interchangeable radio front-end (RF front-end). The radio front-end is a hardware module that receives and/or transmits radio signals through an antenna. In the Sora architecture, the RF front-end represents the well-defined interface between the digital and analog domains. It contains analog-to-digital (A/D) and digital-to-analog (D/A) converters, and necessary circuitry for radio transmission. Since all signal processing is done in software, the RF front-end design can be rather generic. It can be implemented in a self-contained module with a standard interface to the RCB. Multiple wireless technologies defined on the same frequency band can use the same RF front-end hardware, and the RCB can connect to different RF front-ends designed for different frequency bands.

The RCB is a new PC interface board for establishing a high-throughput, low-latency path for transferring high-fidelity digital signals between the RF front-end and PC memory. To achieve the required system throughput discussed in Section 2.1, the RCB uses a high-speed, low-latency bus such as PCIe. With a maximum throughput of 64Gbps (PCIe $\times$ 32) and sub-microsecond latency, it is well suited for supporting multiple gigabit data rates for wireless signals over a very wide band or over many MIMO channels. Further, the PCIe interface is now common in contemporary commodity PCs.

Another important role of the RCB is to bridge the synchronous data transmission at the RF front-end and the asynchronous processing on the host CPU. The RCB uses various buffers and queues, together with a large onboard memory, to convert between synchronous and asynchronous streams and to smooth out bursty transfers between the RCB and host memory. The large onboard memory further allows caching precomputed waveforms, adding additional flexibility for software radio processing.

Finally, the RCB provides a low-latency control path for software to control the RF front-end hardware and to ensure it is properly synchronized with the host CPU. Section 5.1 describes our implementation of the RCB in more detail.

3.2. Sora software

Figure 3 illustrates Sora's software architecture. The software components in Sora provide necessary system services and programming support for implementing various wireless PHY and MAC protocols in a general-purpose operating system. In addition to facilitating the interaction with the RCB, Sora provides a set of techniques to greatly improve the performance of PHY and MAC processing on GPPs. To meet the processing and real-time requirements, these techniques make full use of various common features in existing multi-core CPU architectures, including the extensive use of LUTs, substantial data-parallelism with CPU SIMD extensions, the efficient partitioning of streamlined processing over multiple cores, and exclusive dedication of cores for software radio tasks. We describe these software techniques in details in the next section.

4. HIGH-PERFORMANCE SDR SOFTWARE

4.1. Efficient PHY processing

In a memory-for-computation trade-off, Sora relies upon the large-capacity, high-speed cache memory in GPPs to accelerate PHY processing with precalculated LUTs. Contemporary modern CPU architectures usually have megabytes of L2 cache with a low (10–20 cycles) access latency. If we precalculate LUTs for a large portion of PHY algorithms, we can greatly reduce the computational requirement for online processing.

For example, the *soft demapper* algorithm used in demodulation needs to calculate the *confidence level* of each bit contained in an incoming symbol. This task involves rather complex computation proportional to the modulation density. More precisely, it conducts an extensive search for all modulation points in a constellation graph and calculates a ratio between the minimum of Euclidean distances to all points representing one and the minimum of distances to all points representing zero. In this case, we can precalculate the confidence levels for all possible incoming symbols based on their I and Q values, and build LUTs to directly map the input symbol to confidence level. Such LUTs are not large. For example, in 802.11a/g with a 54Mbps modulation rate (64-QAM), the size of the LUT for the soft demapper is only 1.5KB.

As we detail later in Section 5.2.1, more than half of the common PHY algorithms can indeed be rewritten with LUTs, each with a speedup from 1.5× to 50×. Since the size of each LUT is sufficiently small, the sum of all LUTs in a processing path can easily fit in the L2 caches of contemporary GPP cores. With *core dedication* (Section 4.3), the possibility of cache collisions is very small. As a result, these LUTs are almost always in caches during PHY processing.

To accelerate PHY processing with data-level parallelism, Sora heavily uses the SIMD extensions in modern GPPs, such as SSE, 3DNow! and AltiVec. Although these extensions were designed for multimedia and graphics applications, they also match the needs of wireless signal processing very well because many PHY algorithms have fixed computation structures that can easily map to large vector operations.

4.2. Multi-core streamline processing

Even with the above optimizations, a single CPU core may not have sufficient capacity to meet the processing requirements of high-speed wireless communication technologies. As a result, Sora must be able to use more than one core in a multi-core CPU for PHY processing. This multi-core technique should also be scalable because the signal processing algorithms may become increasingly more complex as wireless technologies progress.

As discussed in Section 2, PHY processing typically contains several functional blocks in a pipeline. These blocks differ in processing speed and in input/output data rates and units. A block is only *ready* to execute when it has sufficient input data from the previous block. Therefore, a key issue is how to schedule a functional block on multiple cores when it is ready.

Sora chooses a static scheduling scheme. This decision is based on the observation that the schedule of each block in a PHY processing pipeline is actually static: the processing pattern of previous blocks can determine whether a subsequent block is ready or not. Sora can thus partition the whole PHY processing pipeline into several sub-pipelines and statically assign them to different cores. Within one sub-pipeline, when a block has accumulated enough data for the next block to be ready, it explicitly schedules the next block. Adjacent sub-pipelines are still connected with a synchronized FIFO (SFIFO), but the number of SFIFOs and their overhead are greatly reduced.

4.3. Real-time support

SDR processing is a time-critical task that requires strict guarantees of computational resources and hard real-time deadlines. As an alternative to relying upon the full generality of real-time operating systems, we can achieve real-time guarantees by simply dedicating cores to SDR processing in a multi-core system. Thus, sufficient computational resources can be guaranteed without being affected by other concurrent tasks in the system.

This approach is particularly plausible for SDR. First, wireless communication often requires its PHY to constantly monitor the channel for incoming signals. Therefore, the PHY processing may need to be active all the time. It is much better to always schedule this task on the same core to minimize overhead like cache misses or TLB flushes. Second, previous work on multi-core OSes also suggests that isolating applications into different cores may have better performance compared to symmetric scheduling, since an effective use of cache resources and a reduction in locks can outweigh dedicating cores.[9] Moreover, a *core dedication* mechanism is much easier to implement than a real-time scheduler, sometimes even without modifying an OS kernel. For example, we can simply raise the priority of a kernel thread so that it is pinned on a core and it exclusively runs until termination (Section 5.2.3).

5. IMPLEMENTATION

5.1. Hardware

We have designed and implemented the Sora RCB as shown in Figure 4. It contains a Virtex-5 FPGA, a PCIe-×8 interface, and 256MB of DDR2 SDRAM. The RCB can connect to various RF front-ends. In our experimental prototype, we use a third-party RF front-end that is capable of transmitting and receiving a 20 MHz channel at 2.4 or 5 GHz.

Figure 5 illustrates the logical components of the Sora hardware platform. The DMA and PCIe controllers interface with the host and transfer digital samples between the RCB and PC memory. Sora software sends commands and reads RCB states through RCB registers. The RCB uses its onboard SDRAM as well as small FIFOs on the FPGA chip to bridge data streams between the CPU and RF front-end. When receiving, digital signal samples are buffered in on-chip FIFOs and delivered into PC memory when they fit

Figure 4. Sora radio control board.

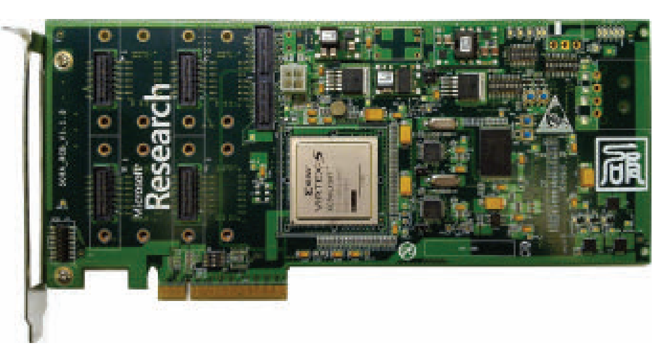
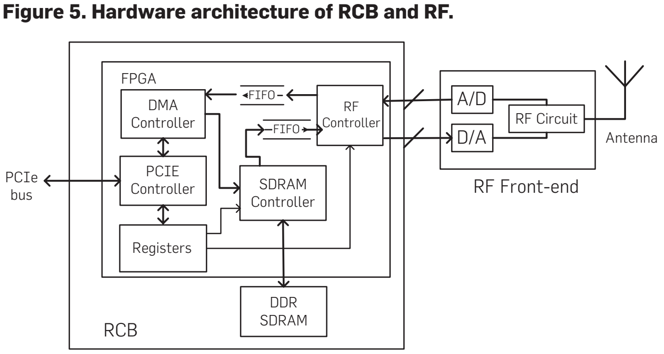


Figure 5. Hardware architecture of RCB and RF.



in a DMA burst (128B). When transmitting, the large RCB memory enables Sora software to first write the generated samples onto the RCB, and then trigger transmission with another command to the RCB. This functionality provides flexibility to the Sora software for precalculating and storing several waveforms before actually transmitting them, while allowing precise control of the timing of the waveform transmission.

While implementing Sora, we encountered a consistency issue in the interaction between DMA operations and the CPU cache system. When a DMA operation modifies a memory location that has been cached in the L2 cache, it does not invalidate the corresponding cache entry. When the CPU reads that location, it can therefore read an incorrect value from the cache.

We solve this problem with a *smart-fetch* strategy, enabling Sora to maintain cache coherency with DMA memory without drastically sacrificing throughput if disabling cached accesses. First, Sora organizes DMA memory into small slots, whose size is a multiple of a cache line. Each slot begins with a *descriptor* that contains a flag. The RCB sets the flag after it writes a full slot of data, and clears it after the CPU processes all data in the slot. When the CPU moves to a new slot, it first reads its descriptor, causing a whole cache line to be filled. If the flag is set, the data just fetched is valid and the CPU can continue processing the data. Otherwise, the RCB has not updated this slot with new data. Then, the CPU explicitly flushes the cache line and repeats reading the same location. This next read refills the cache line, loading the most recent data from memory.

Table 1 summarizes the RCB throughput results, which agree with the hardware specifications. To precisely measure PCIe latency, we instruct the RCB to read a memory address in host memory, and measure the time interval between issuing the request and receiving the response in hardware. Since each *read* involves a round trip operation, we use half of the measured time to estimate the one-way delay. This one-way delay is 360 ns with a worst case variation of 4 ns.

5.2. Software

The Sora software is written in C, with some assembly for performance-critical processing. The entire Sora software is implemented on Windows XP as a network device driver and it exposes a virtual Ethernet interface to the upper TCP/IP stack. Since any software radio implemented on Sora can appear as a normal network device, all existing network applications can run unmodified on it.

PHY Processing Library: In the Sora PHY processing library, we extensively exploit the use of look-up tables (LUTs) and SIMD instructions to optimize the performance of PHY algorithms. We have been able to rewrite more than half of the PHY algorithms with LUTs. Some LUTs are straightforward precalculations, others require more sophisticated implementations to keep the LUT size small. For the soft-demapper example mentioned earlier, we can greatly reduce the LUT size (e.g., 1.5KB for the 802.11a/g 54Mbps modulation) by exploiting the symmetry of the algorithm. In our SoftWiFi implementation described below, the overall size of the LUTs is around 200KB for 802.11a/g and 310KB for 802.11b, both of which fit comfortably within the L2 caches of commodity CPUs.

Table 1. DMA throughput performance of the RCB.

Mode	Rx (Gbps)	Tx (Gbps)
PCIe-x4	6.71	6.55
PCIe-x8	12.8	12.3

We also heavily use SIMD instructions in coding Sora software. We currently use the SSE2 instruction set designed for Intel CPUs. Since the SSE registers are 128-bit wide while most PHY algorithms require only 8-bit or 16-bit fixed-point operations, one SSE instruction can perform 8 or 16 simultaneous calculations. SSE also has rich instruction support for flexible data permutations, and most PHY algorithms, e.g., FFT, FIR Filter and Viterbi, can fit naturally into this SIMD model. For example, the Sora Viterbi decoder uses only 40 cycles to compute the branch metric and select the shortest path for each input. As a result, our Viterbi implementation can handle 802.11a/g at the 54Mbps modulation with only one 2.66 GHz CPU core, whereas previous implementations relied on hardware implementations. Note that other GPP architectures, like AMD and PowerPC, have very similar SIMD models and instruction sets, and we expect that our optimization techniques will directly apply to these other GPP architectures as well.

Table 2 summarizes some key PHY processing algorithms we have implemented in Sora, together with the optimization techniques we have applied. The table also compares the performance of a conventional software implementation (e.g., a direct translation from a hardware implementation) and the Sora implementation with the LUT and SIMD optimizations.

Lightweight, Synchronized FIFOs: Sora allows different PHY processing blocks to streamline across multiple cores, and we have implemented a lightweight, synchronized FIFO to connect these blocks with low contention overhead. The idea is to augment each data slot in the FIFO with a header that indicates whether the slot is empty or not. We pad each data slot to be a multiple of a cache line. Thus, the consumer is always chasing the producer in the circular buffer for filled slots. If the speed of the producer and consumer is the same and the two pointers are separated by a particular offset (e.g., two cache lines in the Intel architecture), no cache miss will occur during synchronized streaming since the local cache will prefetch the following slots before the actual access. If the producer and the consumer have different processing speeds, e.g., the reader is faster than the writer, then eventually the consumer will wait for the producer to release a slot. In this case, each time the producer writes to a slot, the write will cause a cache miss at the consumer. But the producer will not suffer a miss since the next free slot will be prefetched into its local cache. Fortunately, such cache misses experienced by the consumer will not cause significant impact on the overall performance of the streamline processing since the consumer

Table 2. Key algorithms in IEEE 802.11b/a and their performance with conventional and Sora implementations.

Algorithm	Configuration	I/O Size (bit) Input	Output	Optimization Method	Computation Required (Mcycles/s) Conventional Implementation	Sora Implementation	Speedup
IEEE 802.11b							
Scramble	11Mbps	8	8	LUT	96.54	10.82	8.9×
Descramble	11Mbps	8	8	LUT	95.23	5.91	16.1×
Mapping and spreading	2Mbps, DQPSK	8	44 × 16 × 2	LUT	128.59	73.92	1.7×
CCK modulator	5Mbps, CCK	8	8 × 16 × 2	LUT	124.93	81.29	1.5×
	11Mbps, CCK	8	8 × 16 × 2	LUT	203.96	110.88	1.8×
FIR filter	16-bit I/Q, 37 taps, 22MSps	16 × 2 × 4	16 × 2 × 4	SIMD	5,780.34	616.41	9.4×
Decimation	16-bit I/Q, 4× Oversample	16 × 2 × 4 × 4	16 × 2 × 4	SIMD	422.45	198.72	2.1×
IEEE 802.11a							
FFT/IFFT	64 points	64 × 16 × 2	64 × 16 × 2	SIMD	754.11	459.52	1.6×
Conv. encoder	24Mbps, 1/2 rate	8	16	LUT	406.08	18.15	22.4×
	48Mbps, 2/3 rate	16	24	LUT	688.55	37.21	18.5×
	54Mbps, 3/4 rate	24	32	LUT	712.10	56.23	12.7×
Viterbi	24Mbps, 1/2 rate	8 × 16	8	SIMD+LUT	68,553.57	1,408.93	48.7×
	48Mbps, 2/3 rate	8 × 24	16	SIMD+LUT	117,199.6	2,422.04	48.4×
	54Mbps, 3/4 rate	8 × 32	24	SIMD+LUT	131,017.9	2,573.85	50.9×
Soft demapper	24Mbps, QAM 16	16 × 2	8 × 4	LUT	115.05	46.55	2.5×
	54Mbps, QAM 64	16 × 2	8 × 6	LUT	255.86	98.75	2.4×
Scramble and descramble	54Mbps	8	8	LUT	547.86	40.29	13.6×

is not the bottleneck element.

Real-Time Support: Sora uses *exclusive threads* (or *ethreads*) to dedicate cores for real-time SDR tasks. Sora implements ethreads without any modification to the kernel code. An ethread is implemented as a kernel-mode thread, and it exploits the *processor affiliation* that is commonly supported in commodity OSes to control on which core it runs. Once the OS has scheduled the ethread on a specified physical core, it will raise its IRQL (interrupt request level) to a level as high as the kernel scheduler, e.g., dispatch_level in Windows. Thus, the ethread takes control of the core and prevents itself from being preempted by other threads.

Running at such an IRQL, however, does not prevent the core from responding to hardware interrupts. Therefore, we also constrain the *interrupt affiliations* of all devices attached to the host. If an ethread is running on one core, all interrupt handlers for installed devices are removed from the core, thus prevent the core from being interrupted by hardware. To ensure the correct operation of the system, Sora always ensures core zero is able to respond to all hardware interrupts. Consequently, Sora only allows ethreads to run on cores whose ID is greater than zero.

6. EXPERIENCE

To demonstrate the use of Sora, we have developed two wireless systems fully in software in a multi-core PC, namely SoftWiFi and SoftLTE. The performance we report for SoftWiFi is measured on an Intel Core Duo 2 (2.67 GHz), and the performance reported for SoftLTE is measured on an Intel Core i7-920 (2.67 GHz).

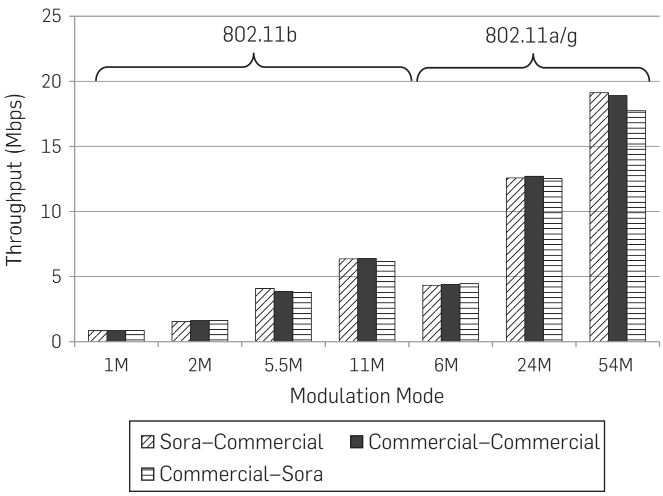
6.1. SoftWiFi

SoftWiFi implements the basic access mode of 802.11. The MAC state machine (SM) is implemented as an ethread. Since 802.11 is a simplex radio, the demodulation components can run directly within a MAC SM thread. If a single core is insufficient for all PHY processing (e.g., 802.11a/g), the PHY processing can be partitioned across two ethreads. These two ethreads are streamed using a synchronized FIFO. Two additional auxiliary threads modulate the outgoing frames in the background and transfer the demodulated frames to upper layers, respectively.

In idle state, the SM continuously measures the average energy to determine whether the channel is clean or there is an incoming frame. If it detects a high energy, SoftWiFi starts to demodulate a frame. After successfully receiving a frame, the 802.11 MAC standard requires a station to transmit an ACK frame in a timely manner (10 μs for 802.11b and 16 μs for 802.11a). This ACK requirement is quite difficult for an SDR implementation in software on a PC. Both generating and transferring the waveform across the PC bus will cause a latency of several microseconds, and the sum is usually larger than mandated by the standard.

Fortunately, an ACK frame generally has a fixed pattern with only a few dynamic fields (i.e., sender address). Thus, we can precalculate most of an ACK frame (19B), and update only the address (10B) on the flight. We can further do it immediately after demodulating the MAC header, and without waiting for the end of a frame. We then prestore the waveform in the memory of the RCB. Thus, the time for ACK

Figure 6. Throughput of Sora when communicating with a commercial WiFi card. *Sora–Commercial* presents the transmission throughput when a Sora node sends data. *Commercial–Sora* presents the throughput when a Sora node receives data. *Commercial–Commercial* presents the throughput when a commercial NIC communicates with another commercial NIC.



generation and transferring can overlap with the demodulation of the data frame. After the entire frame is demodulated and validated, SoftWiFi instructs the RCB to transmit the ACK which has already been stored in the RCB. Thus, the latency for ACK transmission is very small.

Figure 6 shows the transmitting and receiving throughput of a Sora SoftWiFi node when it communicates with a commercial WiFi NIC. In the "Sora–Commercial" configuration, the Sora node acts as a sender and generates 1400-byte UDP frames and unicast transmits them to a laptop equipped with a commercial NIC. In the "Commercial–Sora" configuration, the Sora node acts as a receiver, and the laptop generates the same workload. The "Commercial–Commercial" configuration shows the throughput when both sender and receiver are commercial NICs. In all configurations, the hosts were at the same distance from each other and experienced very little packet loss. Figure 6 shows the throughput achieved for all configurations with the various modulation modes in 11a/b/g. We show only three selective rates in 11a/g for conciseness. The results are averaged over five runs (the variance was very small).

We make a number of observations from these results. First, the Sora SoftWiFi implementation operates seamlessly with commercial devices, showing that Sora SoftWiFi is protocol compatible. Second, Sora SoftWiFi can achieve similar performance as commercial devices. The throughputs for both configurations are essentially equivalent, demonstrating that SoftWiFi (1) has the processing capability to demodulate all incoming frames at full modulation rates, and (2) it can meet the 802.11 timing constraints for returning ACKs within the delay window required by the standard. We note that the maximal achievable application throughput for 802.11 is less than 80% of the PHY data rate, and the percentage decreases as the PHY data rate increases. This limit is due to the overhead of headers at different layers as well as the MAC overhead to coordinate channel access (i.e., carrier sense, ACKs, and backoff), and is a well-known property of 802.11 performance.

6.2. SoftLTE

We have also implemented the 3GPP LTE Physical Uplink Shared Channel (PHUSC) on the Sora platform.[13] LTE is the next generation cellular standard. It is more complex than 802.11 since it uses a higher-order FFT (1024-point) and advanced coding/decoding algorithms (e.g., Turbo coding). Our SoftLTE implementation on Sora provides a peak data rate of 43.8Mbps with a 20-MHz channel, 16QAM modulation, and 3/4 Turbo coding. The most computationally intensive component of an LTE PHY is the Turbo decoder. Our current implementation can achieve 35Mbps throughput using one hardware thread of an Intel Core i7-920 core (2.66 GHz). Since Core i7 supports hyperthreading, though, we can execute the Turbo decoder in parallel on two threads, achieving an aggregated throughput of 54.8Mbps. We can achieve this performance because Turbo decoding is relatively balanced in the number of arithmetic instructions and memory accesses. Therefore, the two threads can overlap these two kinds of operations well and yield a 56% performance gain even though they share the same execution units of a single core. Thus, the whole SoftLTE implementation can run in real time with two Intel Core i7 cores.

7. RELATED WORK

Traditionally, device drivers have been the primary software mechanism for changing wireless functionality on general-purpose computing systems. For example, the MadWiFi drivers for cards with Atheros chipsets,[3] HostAP drivers for Prism chipsets,[2] and the rtx200 drivers for RaLink chipsets[5] are popular driver suites for experimenting with 802.11. These drivers typically allow software to control a wide range of 802.11 management tasks and non-time-critical aspects of the MAC protocol, and allow software to access some device hardware state and exercise limited control over device operation (e.g., transmission rate or power). However, they do not allow changes to fundamental aspects of 802.11 like the MAC packet format or any aspects of PHY.

SoftMAC goes one step further to provide a platform for implementing customized MAC protocols using inexpensive commodity 802.11 cards.[17] Based on the MadWiFi drivers and associated open-source hardware abstraction layers, SoftMAC takes advantage of features of the Atheros chipsets to control and disable default low-level MAC behavior. SoftMAC enables greater flexibility in implementing nonstandard MAC features, but does not provide a full platform for SDR. With the separation of functionality between driver software and hardware firmware on commodity devices, time critical tasks and PHY processing remain unchangeable.

GNU Radio is a popular software toolkit for building software radios using general-purpose computing platforms.[1] GNU Radio consists of a software library and a hardware platform. Developers implement software radios by composing modular precompiled components into processing graphs using Python scripts. The default GNU Radio platform is the Universal Software Radio Peripheral (USRP), a configurable FPGA radio board that connects to the host. As with Sora, GNU Radio performs much of the SDR processing on the host itself. Current USRP supports USB2.0 and a new version USRP 2.0 upgrades to Gigabit Ethernet. Such interfaces, though, are not sufficient for high-speed wireless protocols in wide bandwidth channels. Existing USRP/GNU Radio platforms can only sustain low-speed wireless communication due to both the hardware constraints as well as software processing.[18] As a consequence, users must sacrifice radio performance for its flexibility.

The WARP hardware platform provides a high-performance SDR platform.[8] Based on Xilinx FPGAs and PowerPC cores, WARP allows full control over the PHY and MAC layers and supports customized modulations up to 36Mbps. A variety of projects have used WARP to experiment with new PHY and MAC features, demonstrating the impact a high-performance SDR platform can provide. KUAR is another SDR development platform.[15] Similar to WARP, KUAR mainly uses Xilinx FPGAs and PowerPC cores for signal processing. But it also contains an embedded PC as the control processor host (CPH), enabling some communication systems to be implemented completely in software on the CPH. Sora provides the same flexibility and performance as hardware-based platforms, like WARP, but it also provides a familiar and powerful programming environment with software portability at a lower cost.

The SODA architecture represents another point in the SDR design space.[14] SODA is an application domain-specific multiprocessor for SDR. It is fully programmable and targets a range of radio platforms—four such processors can meet the computational requirements of 802.11a and W-CDMA. Compared to WARP and Sora, as a single-chip implementation it is more appropriate for embedded scenarios. As with WARP, developers must program to a custom architecture to implement SDR functionality.

8. CONCLUSION

This paper presented Sora, a fully programmable software radio platform on commodity PC architectures. Sora combines the performance and fidelity of hardware SDR platforms with the programmability of GPP-based SDR platforms. Using the Sora platform, we also present the design and implementation of SoftWiFi, a software implementation of the 802.11a/b/g protocols, and SoftLTE, a software implementation of the LTE uplink PHY.

The flexibility provided by Sora makes it a convenient platform for experimenting with novel wireless protocols. In our research group, we have extensively used Sora to implement and evaluate various ideas in our wireless research projects. For example, we have built a spatial multiplexing system with 802.11b.[19] In this work, we implemented not only a complex PHY algorithm with successive interference cancellation, but also a sophisticated carrier-counting multi-access (CCMA) MAC—implementations would not have been possible with previous PC-based software radio platforms.

Sora is now available for academic use as the MSR Software Radio Kit.[4] The Sora hardware can be ordered from a vender company in Beijing and all software can be downloaded for free from Microsoft Research website. Our hope is that Sora can substantially contribute to the adoption of SDR for wireless networking experimentation and innovation.

Acknowledgments

The authors would like to thank Xiongfei Cai, Ningyi Xu, and Zenlin Xia in the Hardware Computing group at MSRA for their essential assistance in the hardware design of the RCB. We also thank Fan Yang and Chunyi Peng in the Wireless Networking (WN) Group at MSRA; in particular we have learned much from their early study on accelerating 802.11a using GPUs. We would also like to thank all members in the WN Group and Zheng Zhang for their support and feedback. The authors also want to thank Songwu Lu, Frans Kaashoek, and MSR colleagues (Victor Bahl, Ranveer Chandra, etc.) for their comments on earlier drafts of this paper.

References

1. GNU Radio. http://www.gnu.org/software/gnuradio/.
2. HostAP. http://hostap.epitest.fi/.
3. MadWifi. http://sourceforge.net/projects/madwifi.
4. Microsoft Research Software Radio Platform. http://research.microsoft.com/enus/projects/sora/academickit.aspx.
5. Rt2x00. http://rt2x00.serialmonkey.com.
6. Small Form Factor SDR Development Platform. http://www.xilinx.com/products/devkits/SFF-SDR-DP.htm.
7. Universal Software Radio Peripheral. http://www.ettus.com/.
8. WARP: Wireless Open Access Research Platform. http://warp.rice.edu/trac.
9. Boyd-Wickizer, S., Chen, H., Chen, R., Mao, Y., Kaashoek, F., Morris, R., Pesterev, A., Stein, L., Wu, M., Dai, Y., Zhang, Y., Zhang Z. Corey: an operating system for many cores. In *OSDI* 2008.
10. Cummings, M., Haruyama, S. FPGA in the Software Radio. *IEEE Commun. Mag.* 1999.
11. de Vegte, J.V. *Fundamental of Digital Signal Processing*. Cambridge University Press, 2005.
12. Glossner, J., Hokenek, E., Moudgill, M. The Sandbridge Sandblaster Communications Processor. In *3rd Workshop on Application Specific Processors* (2004).
13. Li, Y., Fang, J., Tan, K., Zhang, J., Cui, Q., Tao, X. Soft-LTE: a software radio implementation of 3GPP long term evolution based on Sora platform. In *ACM Moicom 2009 (Demonstration)* (Beijing, 2009).
14. Lin, Y., Lee, H., who, M., Harel, Y., Mahlke, S., Mudge, T. SODA: a low-power architecture for software radio. In *ISCA '06: Proceedings of the 33rd International Symposium on Computer Architecture* (2006).
15. Minden, G.J., Evans, J.B., Searl, L., DePardo, D., Patty, V.R., Rajbanshi, R., Newman, T., Chen, Q., Weidling, F., Guffey, J., Datla, D., Barker, B., Peck, M., Cordill, B., Wyglinski, A.M., Agah, A. KUAR: a flexible software-defined radio development platform. In *DySpan* (2007).
16. Neel, J., Robert, P., Reed, J. A formal methodology for estimating the feasible processor solution space for a software radio. In *SDR'05: Proceedings of the SDR Technical Conference and Product Exposition* (2005).
17. Neufeld, M., Fifield, J., Doerr, C., Sheth, A., Grunwald, D. SoftMAC—flexible wireless research platform. In *HotNets'05* (2005).
18. Schmid, T., Sekkat, O., Srivastava, M.B. An experimental study of network performance impact of increased latency in software defined radios. In *WiNETCH'07* (2007).
19. Tan, K., Liu, H., Fang, J., Wang, W., Zhang, J., Chen, M., Voelker, G.M. SAM: enabling practical spatial multiple access in wireless LAN. In *MobiCom'09: Proceedings of the 15th Annual International Conference on Mobile Computing and Networking* (New York, NY, 2009), ACM, USA, 49–60.

Kun Tan (kuntan@microsoft.com), Microsoft Research Asia, Beijing, China.

He Liu (h8liu@ucsd.edu), University of California, San Diego, La Jolla, CA.

Jiansong Zhang (kuntan@microsoft.com), Microsoft Research Asia, Beijing, China.

Yongguang Zhang (ygz@microsoft.com), Microsoft Research Asia, Beijing, China.

Ji Fang (v-fangji@microsoft.com), Microsoft Research Asia and Beijing Jiaotong University, Beijing, China.

Geoffrey M. Voelker (voelker@cs.ucsd.edu), University of California, San Diego, La Jolla, CA.

DOI:10.1145/1866739.1866761

Technical Perspective
Multipath: A New Control Architecture for the Internet

By Damon Wischik

MULTIPATH TRANSMISSION FOR the Internet—that is, allowing users to send some of their packets along one path and others along different paths—is an elegant solution still looking for the right problem.

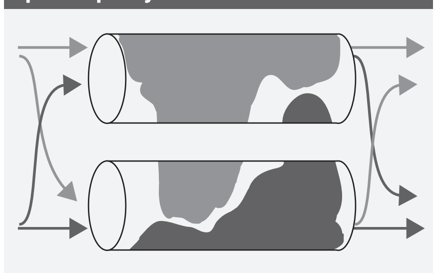
The most obvious benefit of multipath transmission is greater reliability. For example, I'd like my phone to use WiFi when it can, but seamlessly switch to cellular when needed, without disrupting my flow of data. In general, the only way to create a reliable network out of unreliable components is through redundancy, and multipath transmission is an obvious solution.

The second benefit of multipath transmission is that it gives an extra degree of flexibility in sharing the network. Just as packet switching removed the artificial constraints imposed by splitting links into circuits, so too multipath removes the artificial constraints imposed by 'splitting' the network's total capacity into separate links (see the accompanying figure).

Flexibility comes with dangers. By building the Internet with packet switching, we no longer had the control over congestion that circuit switching provides (crude though it may be), and this led in 1988 to Internet congestion collapse. Van Jacobson[3] realized there needed to be a new system for controlling congestion, and he had the remarkable insight that it could be achieved by end systems on their own. The Internet has been using his transmission control protocol (TCP) largely unchanged until recently.

The flexibility offered by multipath transport also brings dangers. The claim of the following paper is that, once we do away with the crude control of "each flow may use only one path," there should be some new control put in place—and, in fact, the proper control can be achieved by end systems on their own. That is to say, if multipath is packet switching 2.0, then it needs TCP 2.0.

If flows have access to multiple paths, then spikes in traffic on one link can make use of spare capacity on other links.



Internet congestion collapse in 1988 was a powerful motivator for the quick deployment of Jacobson's TCP. There is not yet a killer problem for which multipath congestion control is the only good solution. Perhaps we will be unlucky enough to find one. (It has been shown[1] that simple greedy route choice by end users, combined with intelligent routing by network operators, can in theory lead to arbitrarily inefficient outcomes, but this has not been seen in practice.)

Lacking a killer problem, the authors present four vignettes that illustrate the inefficiency and unfairness of a naïve approach to multipath, and that showcase the benefit of clever multipath congestion control.

The niggling problems of naïve approaches to multipath could probably all be mitigated by special-case fixes such as "only use paths whose round trip times are within a factor of two of each other" or "no flow may use more than four paths at a time," perhaps enforced by deep packet inspection. So, in effect, the authors present a choice between a single clean control architecture for multipath transmission, and a series of special-case fixes.

The naïve approach to multipath, as studied in this paper, is to simply run separate TCP congestion control on each path. The clever alternative is to couple the congestion control on different paths, with the overall effect of shifting traffic away from more-congested paths onto less-congested paths; two research groups[2,4] have independently devised an appropriate form of coupling. This is the approach under exploration in the mptcp working group at the IETF, although with some concessions to graceful coexistence with existing TCP.

The differences between the two sorts of congestion control show up both in the overall throughput of the network, and also in how the network's capacity is allocated. The authors use the framework of social welfare utility maximization to address both metrics in a unified way. This framework has been mainstream in theoretical research on congestion control for the past decade. But it is not mainstream in systems work, where more intuitive metrics such as average throughput and Jain's fairness index hold sway, along with views like "Congestion is only a problem at access links, and if I've paid for two connections then I ought to be able to use two TCP flows." These differences in language and culture have meant that the paper's conclusions have not become systems orthodoxy.

Now that multipath transport protocols are a hot topic in the network systems community, it is a good time to highlight this work, and to translate its conclusions into practical answers about systems such as data centers and multihomed mobile devices. The authors only address congestion control and path selection for an idealized model of moderately long-lived flows. There are still important questions to answer, such as: When is a flow long enough to make it worth opening a new path? When is a path so bad it should be closed?

References

1. Acemoglu, D., Johari, R. and Ozdaglar, A.E. Partially optimal routing. *IEEE Journal of Selected Areas in Communication 25* (2007), 1148–1160.
2. Han, H., Shakkottai, S., Hollot, C.V., Srikant, R. and Towsley, D.F. Multi-path TCP: A joint congestion control and routing scheme to exploit path diversity in the Internet. *IEEE/ACM Transactions on Networking 16* (2006), 1260–1271.
3. Jacobson, V. Congestion avoidance and control. In *Proceedings of SIGCOMM 1988 Conference.*
4. Kelly, F.P. and Voice, T. Stability of end-to-end algorithms for joint routing and rate control. *ACM/SIGCOMM Computer Communication Review 35* (2005), 5–12.

Damon Wischik (d.wischik@cs.ucl.ac.uk) is a Royal Society university research fellow in the Networks Research Group in the Department of Computer Science at University College London.

DOI:10.1145/1866739.1866762

Path Selection and Multipath Congestion Control

By Peter Key, Laurent Massoulié, and Don Towsley

Abstract

In this paper, we investigate the benefits that accrue from the use of multiple paths by a session coupled with rate control over those paths. In particular, we study data transfers under two classes of multipath control, *coordinated control* where the rates over the paths are determined as a function of all paths, and *uncoordinated control* where the rates are determined independently over each path. We show that coordinated control exhibits desirable load balancing properties; for a homogeneous static random paths scenario, we show that the worst-case throughput performance of uncoordinated control behaves as if each user has but a single path (scaling like log(log(*N*))/log(*N*) where *N* is the system size, measured in number of resources), whereas coordinated control yields a worst-case throughput allocation bounded away from zero. We then allow users to change their set of paths and introduce the notion of a Nash equilibrium. We show that both coordinated and uncoordinated control lead to Nash equilibria corresponding to desirable welfare maximizing states, provided in the latter case, the rate controllers over each path do not exhibit any round-trip time (RTT) bias (unlike TCP Reno). Finally, we show in the case of coordinated control that more paths are better, leading to greater welfare states and throughput capacity, and that simple path reselection polices that shift to paths with higher net benefit can achieve these states.

1. INTRODUCTION

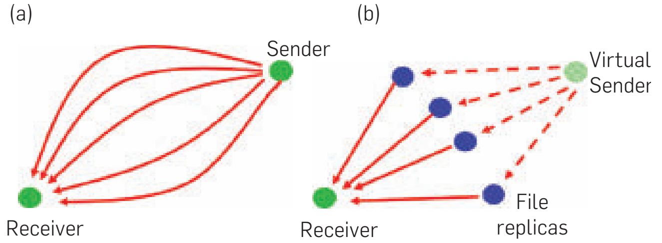
Multipath routing has received attention recently.[2, 5, 6, 14, 21] Furthermore, combining multipath routing with rate control is implicitly used by several peer-to-peer (P2P) applications. Most relevant to us is BitTorrent,[4] which maintains a number of, typically four, active connections to other peers with an additional path periodically chosen at random together with a mechanism that retains the best paths (as measured by throughput).

The basic setting of multipath coupled with rate control is as follows. A source and destination pair in a network is given a set of possibly overlapping paths connecting them. The pair then chooses a subset to use and the rates at which to transfer data over those paths. This scenario is illustrated in Figure 1a. Note that the P2P example described above falls into this formulation once one includes a fictitious source feeding data through peers to the intended receiver, as shown in Figure 1b. Some natural questions arise:

- How many paths are required? And does it suffice, as with the above P2P application, to use a subset of the paths? Opening multitudinous TCP connections has negative systems performance implications, hence there are incentives to keep the number of connections small.
- P2P applications use independent *uncoordinated* TCP rate control mechanisms over each active path as this is straightforward to implement and requires no change to the network. However, starting from first principles, mechanism design produces a *coordinated* control mechanism where the rates over each path are determined as a function of all of the paths. How does an uncoordinated control mechanism perform relative to a coordinated control mechanism? This is important because the latter requires a revised transport layer protocol or a careful application layer solution whereas the former is easily implemented using TCP.

The motivating application scenario is of data transfers over a network, where the transfers are long enough to allow performance benefits for multipath routing, although our results apply more generally to situations where there are alternative resources that can help service a demand, and where the demand is serviced using some form of rate control. We assume that demand is fixed, and each user[a] attempts to optimize its performance by choosing appropriate paths (resources), where the rate control algorithm is fixed. More precisely, we assume that the rate control is implicitly characterized by a utility maximization problem,[20] where a particular rate control algorithm

Figure 1. (a) A canonical multipath example. (b) A BitTorrent example where a receiving peer receives data from four peers. A virtual sender has been included to show the relationship to canonical multipath.



[a] We use the term "user" as a convenient shorthand for a user, or the software or algorithm a user or end-system employs.

The original version of this paper was published in the *Proceedings of IEEE Infocom 2007*, May 2007 by IEEE.

(e.g. TCP Reno) maps to a particular (user) utility function,[9] and users selfishly seek to choose paths in such a way as to maximize their net utility. Within this optimization framework, a coordinated controller is modeled by a single utility function per user, whose argument is the aggregate rate summed over paths, whereas an uncoordinated controller has a utility function per path and the aggregation is over all of the utility functions.

Key to the usefulness of multipath rate control is its ability in the hands of users operating independently of each other to balance the load throughout the network. We illustrate this for a particular scenario, where the paths chosen are fixed and static, but chosen at random from a set of size N. We focus on the worst-case allocation, which is a measure of the fairness of the scheme. In the uncoordinated case, the worst-case allocation scales as $\log(\log(N))/\log(N)$ independent of the number b of paths chosen. In contrast, in the coordinated case where each user can balance its load across the b paths available to it, provided there are two or more, the worst-case allocation is bounded away from zero. This demonstrates that

1. Coordinated control balances loads significantly better than uncoordinated when paths are fixed.
2. Coordinated improves on greedy least-loaded resource selection, as in Mitzenmacher,[16] where the least-loaded selection of b resources scales as $1/\log(\log(N))$ for $b > 1$.

Effectively, coordinated control is able to shift the load among the resources, and with each user independently balancing loads over no more than two paths, able to utilize the resources *as if* global load balancing was being performed.

This raises the question of how users should be assigned a set of paths to use. One natural path selection mechanism is to allow users to make their own choices. We study this as a game between users and consider a natural notion of a Nash equilibrium in this context, where users seek to selfishly maximize their own net utilities. We find that when users use coordinated controllers, the Nash equilibria coincide with welfare-maximizing social optima. When we consider uncoordinated controllers, then the results depend on whether the controllers exhibit RTT bias (like TCP) or not. When they do not exhibit RTT bias, the Nash equilibria also coincide with welfare-maximizing social optima. Otherwise they need not.

We show that increasing the number of paths available to a source destination pair is desirable from a performance perspective. However, the simultaneous use of a large number of paths may not be possible. We show that this does not pose a problem as simple path selection policies that combine random path resampling with moving to paths with higher net benefit lead to welfare maximizing equilibria and also increase the throughput capacity of the network. In fact such a policy does as well as if each user uses all of the available paths simultaneously.

In summary, we shall provide some partial answers to our initial questions.

- In a large system, provided users re-select randomly from the sets of paths and shift to paths with higher net benefit, they can rely on a small number of paths and do as well as if they were fully using all available paths.
- Coordinated control has better fairness properties than uncoordinated in the static case. When combined with path reselection, uncoordinated control only does as well as a coordinated control if there is no RTT bias in the controllers.

We conclude the paper with some thoughts on how multipath rate control might be deployed.

2. THE MULTIPATH FRAMEWORK

The standard model of the network is as a capacitated graph $G = (V, E, C)$ where V represents a set of end-hosts or routers, E is a set of communication links and each link has a capacity, say in bits per second, C_l, $l \in E$. In addition a large population of sessions perform data transfers over the network. These sessions are partitioned into a set of session classes $\mathcal{S}$ with N_s sessions in class $s \in \mathcal{S}$. Associated with class s is a source σ_s, a destination d_s, and a set of one or more, possibly overlapping paths, $\mathcal{R}(s)$ between the source and destination that is made available to all class s sessions. Finally, we associate an increasing, concave function with each session class, $U_s(x)$, which is the utility that a class s session receives when it sends data at rate $x > 0$ from source to destination. Now, exactly how this utility is used and the meaning of x depends on whether we are concerned with coordinated or uncoordinated control. We will shortly describe each of these in turn.

A discussion of how utility functions can be used to model standard TCP Reno is given in Kunniyur and Srikant.[15] The so-called weighted alpha-fair utility functions given by

$$U_r(x) = \begin{cases} w_r \dfrac{x^{1-\alpha}}{1-\alpha} & \text{if } \alpha \neq 1 \\ w_r \log(x) & \text{if } \alpha = 1 \end{cases} \tag{1}$$

were introduced in Mo and Walrand,[17] and are linked to different notions of fairness. For example, $\alpha = 1$ corresponds to (weighted) proportional fairness,[8] and $\lim \alpha \to \infty$ to max–min fairness. TCP's behavior is well approximated by taking $\alpha = 2$ and $w_r = 1/T_r^2$, where T_r is the round trip time for path r, in the following sense: TCP achieves the maximum aggregate utility, for given paths and link capacities, for the corresponding utility functions U_r.

The set of paths available to a class s session can potentially be very large. Hence a session will likely use only a small subset of these paths. We assume for now that every class s session uses exactly b_s paths. Let c denote a subset of $\mathcal{R}(s)$ that contains b_s paths and $\mathcal{C}(s)$ the set of all such subsets of paths, $\mathcal{C}(s) = \{c : c \subset \mathcal{R}(s) \wedge |c| = b_s\}$. Let N_c denote the number of class s sessions that use the set of paths $c \in \mathcal{C}(s)$, $s \in \mathcal{S}$, and hence $N_s = \sum_{c \in \mathcal{C}(s)} N_c$. Last, let N_r denote the number of sessions that use path $r \in \mathcal{R}(s)$, $N_r = \sum_{c \in \mathcal{C}(s)} \mathbf{1}(r \in c))N_c$, where $1(x)$ is the indicator function taking the value 1 when x is true.

Associated with each class s session is a congestion controller (rate controller) that determines the rates at which to send data over each of the b_s paths available to it. We now distinguish between coordinated and uncoordinated control.

Coordinated control. Given a set of paths c, a coordinated controller actively balances loads over all paths in c, taking into account the states of the paths. Our understanding of and ability to design such controllers relies on a significant advance made by Kelly et al.,[8] which maps this problem into one of utility optimization. In the case of coordinated congestion control, the objective is to maximize the "social welfare," that is to

$$\text{Maximize} \sum_{s\in\mathcal{S}} \sum_{c\in\mathcal{C}(s)} N_c U_s\Big(\sum_{r\in c} \lambda_{cr}\Big) \quad (2)$$

over ($\lambda_{cr} \geq 0$) subject to the capacity constraints

$$\sum_{r\ni l} \sum_{c\ni r} N_c \lambda_{cr} \leq C_l, \qquad l \in E \quad (3)$$

where λ_{cr} is the sending rate of a class s session that is using path r in $c \in \mathcal{C}(s)$. We will find it useful to represent the total rate contributed by class s sessions that use path $r \in \mathcal{R}(s)$ as $\Lambda_r = N_c \sum_{c\ni r} \lambda_{cr}$, and the aggregate rate achieved by a single s session over all paths in c as $\lambda_c = \sum_{r\in c} \lambda_{cr}$.

Note that in the absence of restrictions on the number of paths used, $\mathcal{C}(s) = \mathcal{R}(s)$, and the optimization can be written

$$\text{Maximize} \sum_{s\in\mathcal{S}} N_s U_s\Big(\sum_{r\in\mathcal{R}(S)} \lambda_r\Big) \quad (4)$$

subject to the capacity constraints. We shall see later in Section 5 that by using random path reselection the solution to (2) actually solves (4), and hence give conditions for when the restriction to using a subset of paths of limited size imposes no performance penalties.

More generally, we can replace the hard capacity constraints[b] by a convex nondecreasing penalty function Γ. In the context of TCP, this penalty function can be thought of as capturing the signaling conveyed by packet losses or packet marking (ECN[19]) by the network to the sessions when link capacities are violated. Under this extension, the coordinated control problem transforms to

$$\text{Maximize} \sum_{S\in\mathcal{S}} \sum_{c\in\mathcal{C}(S)} N_c U_s\Big(\sum_{r\in c} \lambda_{cr}\Big) - \Gamma(\{\Lambda_r\}). \quad (5)$$

There are many ways to approach the problem of designing controllers that solve these problems, but a very natural one is suggested by the TCP congestion control, which solves this variation of the above problem when each session is restricted to a single path (see Key et al.[11]).

Uncoordinated control. As mentioned earlier, uncoordinated control corresponds to a session with path set c executing independent rate controllers over each path in c. This is easily done in the current Internet by establishing separate TCP connections over each path. The ease in which uncoordinated control can be implemented motivates our study of it. In Kelly's optimization formulation this corresponds to solving the following problem:

$$\text{Maximize} \sum_{S\in\mathcal{S}} \sum_{c\in\mathcal{C}(s)} N_c \sum_{r\in c} U_r(\lambda_{cr}) \quad (6)$$

over nonnegative λ_{cr} subject to the capacity constraints (3). As above, by analogy with (5) the constraints can be generalized to reflect the signaling used by a controller such as TCP. Note the difference between this formulation and that for coordinated control. In the case of the latter, the utility is applied to the *aggregate sending rate* whereas in the case of the former, the utility is evaluated on each path and then summed over all paths. Note also that really we have written U_r instead of U_s for the uncoordinated controller, to reflect the fact that the congestion control may differ across different paths (as is the case with TCP whose allocation depends on the RTT of the path).

The functions above are strictly concave and are being optimized over a convex feasible region. Hence the problems admit to unique solutions in terms of aggregate per class rates, even though distinct solutions may exist.

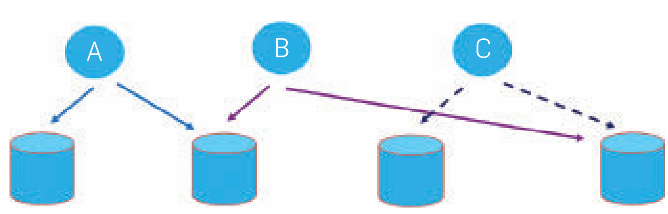
3. LOAD BALANCING PROPERTIES OF MULTIPATH

Multipath has been put forward as a mechanism that when used by all sessions can balance traffic loads in the Internet. It is impossible to determine whether this is universally true. However, we present in this section a simple scenario where this issue can be definitively resolved. We consider a simple scenario where there are N resources with unit capacity ($C_l \equiv 1$).

To provide a concrete interpretation, the resources can be interpreted as servers, or as relay or access nodes—see Figure 2. There are aN users. Each user selects b resources at random from the N available, where b is an integer larger than one (the same resource may be sampled several times). We shall look at the worst-case rate allocation of users in two scenarios. In the first scenario, users implement uncoordinated multipath congestion control where there is no coordination between the b distinct connections of each user. Thus, a connection sharing a resource handling X connections overall achieves a rate allocation of exactly $1/X$. In the second scenario, each user implements coordinated multipath congestion control.

We take the worst-case user rate allocation (or throughput), as the load balance metric. One can show[13] that the more "unfair" the allocation, the greater the expected time to download a unit of data.

Figure 2. Load balancing example: there are *N* servers, *aN* users and each selects *b* > 1 servers at random.



[b] The hard constraints in (3) can be written as the sum of penalty functions, each of which is a step function $\Gamma_l(x)$, with $\Gamma_l(x) = 0$ if $x \leq C_l$ and ∞ otherwise

3.1. Uncoordinated congestion control

Denote by λ_i the total rate that user i obtains from all its connections. In the case of uncoordinated congestion control, we can show that the worst-case rate allocation, $\min \lambda_i$ decreases like $b^2 \log(\log N)/\log N$ as N increases. This is to be compared with the worst-case rate allocation that one gets when $b = 1$, that is when a single path is used: from classical balls and bins models,[16] this also decreases as $\log(\log(N))/\log(N)$ as N increases. It should come as no surprise that using more than two paths exhibits the same asymptotic performance as using only one path; there is no potential for balancing load within the network when all connections operate independent of each other. A formal statement and proof of this result can be found in Key et al.[11]

3.2. Coordinated congestion control

Here we assume as before that there are aN users, each selecting b resources at random, from a collection of N available resources. Denote by λ_{ij} the rate that user i obtains from resource j, and let $\mathcal{R}(i)$ denote the set of resources that user i accesses. In contrast with the previous situation, we now assume that the rates λ_{ij} are chosen to maximize:

$$\sum_{i=1}^{aN} U\Big(\sum_{j\in\mathcal{R}(i)} \lambda_{ij}\Big) \text{ subject to } \sum_i \lambda_{ij} \le 1 \quad \text{for all } j,$$

for some concave utility function U.

An interesting property of this problem is that the set of $\{\lambda^*_{ij}\}$ that solves the above optimization is insensitive to the choice of utility function U so long as it is concave and increasing. Moreover, this insensitivity implies that the optimal aggregate user rates (λ^*_i) correspond to the max–min fair rate allocations (see Bertsekas and Gallager,[3] Section 6.5.2). Simply stated a rate allocation (λ^*_i) is said to be *max–min fair* if and only if an increase of any rate λ^*_{i0} must result in the decrease of some already smaller rate. Formally, for any other feasible allocation (x_i), if $x_i > \lambda^*_i$ then there must exist some j such that $\lambda^*_j < \lambda^*_i$ and $x_j < \lambda^*_j$. The above statements are easily verified by checking that the max–min fair allocation satisfies the Karush–Kuhn–Tucker conditions associated with the above optimization problem.

This leads to the following result.

THEOREM 1. *Assume there are N resources, and aN users each connecting to b resources selected at random. Denote by* $\{\lambda^*_i\}$ *the optimal allocations that result. Then there exists* $x > 0$, *that depends only on a and b, such that:*

$$\lim_{N\to\infty} \mathbf{P}\left(\min_i \lambda^*_i \ge x\right) = 1. \tag{7}$$

The style of the proof has wide applicability and we outline it here: first, an application of Hall's celebrated marriage theorem shows that the minimum allocation will be at least x provided that any set of users (of size n say) connect to at least x times as many servers (nx servers). If this condition is satisfied, the allocation (λ^*_i) will exceed x; hence it is sufficient to ensure that Hall's condition is met with high probability for all nonempty subsets of $1, \ldots, aN$. Using the binomial theorem and the union bound yields an upper bound on the probability the condition fails to hold, and then Stirling's approximation is used to approximate this bound.

This result says that the worst-case rate allocation is bounded away from zero as N tends to infinity, i.e., it is $O(1)$ in the number of resources N. Thus coordinated control exhibits significantly better load balancing properties than does uncoordinated control. It is also interesting to compare this result to the result quoted by Mitzenmacher et al.,[16] which says that if users arrive in some random order, and choose among their b candidate resources one with the lowest load, then the worst-case rate scales like $1/\log(\log(N))$, which unlike the allocation under coordinated control, goes to zero as N increases. The difference between the two schemes is that in Mitzenmacher's scheme a choice has to be made immediately at arrival, which cannot be changed afterward, whereas a coordinated controller actively and adaptively balances load over the b paths reacting to changes that may occur to the loads on the resources.

4. A PATH SELECTION GAME

In this section we address the following question. Suppose that each session is restricted to using exactly b paths each, taken from a much larger set of possible paths: What is the effect of allowing each user to choose its b paths so as to maximize the benefit that it receives? To answer this question, we study a *path selection game*. Here each session is a player that greedily searches for throughput-optimal paths. We characterize the equilibrium allocations that ensue. We show that the same equilibria arise with coordinated congestion control and uncoordinated congestion control provided that the latter does not introduce RTT biases on the different paths. Moreover, these equilibria correspond to the optimal set of rates that solve problems (2) and (6), i.e., achieve welfare maximization. We shall use the models and notation of Section 2.

We shall restrict attention to when N_s is large, so that a change of paths by an individual player (session) does not significantly change the network performance. In game theory terms we are only considering *non-atomic* games.

4.1. Coordinated congestion control

For coordinated control, we use the model of Section 2, where the number of sessions N_s is fixed for all s, and introduce the following notion of a Nash equilibrium.

DEFINITION 1. *The nonnegative variables* N_c, $c \in \mathcal{C}(s)$, $s \in \mathcal{S}$, *are a Nash equilibrium for the coordinated congestion control allocation if they satisfy the constraints* $\sum_c N_c = N_s$, *and, moreover, for all* $s \in \mathcal{S}$, *all* $c \in \mathcal{C}(s)$, *if* $N_c > 0$, *then the corresponding coordinated rate allocations satisfy*

$$\sum_{r\in c} \lambda_{cr} = \max_{c'\in\mathcal{C}(s)} \sum_{r\in c'} \lambda_{c'r}. \tag{8}$$

In other words, for each session (player), weight is only given to sets c that maximize the throughput for s. ◊

We then have the following.

Theorem 2. *At a Nash equilibrium as in Definition 1, the path allocations* λ_r *solve the welfare maximization problem* (4).

The proof follows since at a Nash equilibrium, type s players only use minimum "cost" paths, which can be shown to coincide with the Kuhn–Tucker conditions of (4). This result says that a selfish choice of path sets by end-users results in a solution that is socially optimal.

4.2. Uncoordinated control

We introduce the following notion of Nash equilibrium.

Definition 2. *The collection of per path connection numbers* N_r *is a Nash equilibrium for selfish throughput maximization if it satisfies* $\sum_r N_r = N_s$, *and furthermore, the allocations (6) are such that for all* $s \in \mathcal{S}$, *all* $r \in \mathcal{R}(s)$, *if* $N_r > 0$, *then*

$$\lambda_r = \max_{r' \in \mathcal{R}(s)} \Lambda_{r'}. \tag{9}$$

◊

The intuition for this definition is as follows: any class s session maintains a connection along path r only if it cannot find an alternative path r' along which the default congestion control mechanism would allocate a larger rate.

We then have the following result, whose proof is similar to that of Theorem 2.

Theorem 3. *Assume that for each* $s \in \mathcal{S}$, *there is a class utility function* U_s *such that* $U_r(x) \equiv U_s(x/b)$ *for all* $r \in \mathcal{R}(s)$. *Then for a Nash equilibrium* (N_r), *the corresponding rate allocations* (λ_r) *solve the general optimization problem* (4).

To summarize: if (i) the utility functions associated with the congestion control mechanism are path-independent, (ii) users agree to concurrently use a fixed number b of paths, and (iii) they manage to find throughput-optimal paths, that is they achieve a Nash equilibrium, then at the macroscopic level, the per-class allocations solve the coordinated optimization problem (4).

It is well known that the bandwidth shares achieved by TCP Reno are affected by the path round trip times. Thus the underlying utility functions are necessarily path dependent and the above result does not apply as (i) fails to hold. As a consequence "bad" Nash equilibrium can exist. Indeed, a specific example is given in Key et al.[11] where the preference of TCP for "short-thin links" over "fat-long-links" gives rise to a Nash equilibrium where the throughput is half of what could be achieved. If (ii) is relaxed, different uses have different "market power," where those with larger b_s gain a large share, thus also creating "bad" Nash equilibria in general.

5. MULTIPATH ROUTING WITH RANDOM PATH RESELECTION

In the previous section we explored the effect of allowing users to greedily select their set of paths (b in number) out of the set of all possible paths that are available to them. In this section we focus on two questions. The first regards how many paths, b_s, to allow each class s user so as to enhance its performance and that of the system. We establish a monotonicity result for coordinated control in order to address this question. The second question regards how to manage the overhead that may ensue due to the need for a user to balance load actively over a large number of paths. Possibly surprisingly, we will show that it suffices for a user to maintain a small set of paths, say two ($b = 2$), provided that it repeatedly selects new paths at random and replaces the old paths with these paths when the latter provide higher throughput. It is interesting to point out that BitTorrent uses a strategy much like this where it "unchokes" a peer (tries out a new peer) and replaces the lowest performing of its existing four connections with this new connection if the latter exhibits higher throughput.

We will examine the above questions for both coordinated control and uncoordinated control. We begin with coordinated control.

5.1. Coordinated control

We begin by addressing the first question, namely how many paths are needed. Consider a network G that supports a set of flow classes $\mathcal{S}$ with populations $\{N_s\}$, and utility functions $\{U_s\}$. Let $\{\mathcal{R}(s)\}$ and $\{\mathcal{R}'(s)\}$ be two collections of paths for $\mathcal{S}$ that satisfy $\mathcal{R}(s) \subseteq \mathcal{R}'(s)$ for $s \in \mathcal{S}$ and suppose that each session applies coordinated control over these paths. Then for the problem (4)

$$\sum_{s\in\mathcal{S}} N_s U_s\Big(\sum_{r\in\mathcal{R}(s)} \lambda_{sr}\Big) \le \sum_{s\in\mathcal{S}} N_s U_s\Big(\sum_{r\in\mathcal{R}'(s)} \lambda_{sr}\Big)$$

and hence performance increases when the path sets increase, with performance measured by the optimal welfare under the capacity constraints. This follows from the fact that a solution honoring the constraints on path-sets $\{\mathcal{R}\}$ remains a feasible solution when the set of paths increases.

Remark 1. *Although we have not shown strict inequality, it is easy to construct examples where aggregate utility strictly increases as more and more paths are provided.*

The above result suggests that we would like to provide each user with as large a set of paths possible to perform active load balancing on. However, this comes with the overhead of having to maintain these paths. We examine this issue next by considering a simple policy where a session is given a set of possible paths to draw from, say $\mathcal{R}(s)$ for a class s session, and the policy allows the session to actively use a small subset of these paths, say two of them, while at the same time constantly trying out new paths and replacing poorly performing paths in the old set with better performing paths in the new set. More specifically we consider the following path selection mechanism. Assume a user is using path set c. This user is offered a new path set c' at some fixed rate $A_{cc'}$. This new path set is accepted under the condition that the user receives a higher aggregate rate than it was receiving under c. This process then repeats.

We use the model of Section 2, where the class s users, N_s in number, are divided according to the set of paths they are currently using, $N_c(t)$ denoting the number of class s-users actively using paths in $c \subset \mathcal{R}(s)$ at time t. Class s users actively using the set c of paths consider replacing their path set c with path set c' according to a Poisson process with intensity $A_{cc'}$. We shall assume that $|c| = |c'| = b$, i.e., the number of paths in an active set is fixed at b. Finally, assume that for each class s, any $r \in \mathcal{R}(s)$, any given set $c \in \mathcal{C}(s)$, there is some c' such that $r \in c'$ and $A_{cc'}$ is positive (recall that $\mathcal{C}(s)$ is defined as the collection of size b subsets of $\mathcal{R}(s)$). This assumption states that all paths available to a class s session should be tried no matter what set of initial paths is given to that session.

We also have to concern ourselves with the sending rates of the different users as path reselection proceeds over time. Let $\lambda_c(t)$ denote the data transfer rate for a user actively using path set c, $\lambda_c(t) = \sum_{r\in c} \lambda_{c,r}(t)$ where $\lambda_{c,r}(t)$ is the sending rate along path r at time t. We have described in Key et al.[11] a dynamic process where the vectors $\{N_c(t), \lambda_{c,r}(t)\}$ change over time. This process is stochastic in nature and consequently difficult to model. However, if we assume that the population of users in each class is large, which is reasonable for the Internet, then we can model this process over time by a set of ordinary differential equations, representing the path reselection and rate adaptation dynamics of users over their active path sets. Under the condition that the utility functions and penalty functions are well behaved, we can show that $N_c(t)$ converges to a limit N_c and $\lambda_{cr}(t)$ converges to λ_{cr} as t tends to infinity. Remarkably, we can show that these limits are the maximizers of

$$\mathcal{W}(\lambda, N) := \sum_{s\in\mathcal{S}} \sum_{c\subset\mathcal{R}(s)} N_c U_s(\lambda_c) - \Gamma(\Lambda_r) \qquad (10)$$

subject to $\sum_{c\in\mathcal{C}(s)} N_c = N_s$. In other words, this resampling process allows the system to converge to a state where the proportion of class s sessions using active path set $c \in \mathcal{R}(s)$ and the aggregate rates at which they use these paths maximize the aggregate sum of utilities. This is more precisely stated in the following theorem.

THEOREM 4. *Assume that the utility functions U_s and the penalty function Γ are continuously differentiable on their domain, that the former are strictly concave increasing, and the latter convex increasing. Assume further that $U_s'(x) \to 0$ as $x \to \infty$. Then $(N_c, \lambda_{c,r})$ converges to the set of maximizers of the welfare function (10) under the constraints $\sum_{c\in\mathcal{C}(s)} N_c = N_s$. The corresponding equilibrium rates (λ_r) are solutions of the coordinated welfare maximization problem (2).*

The proof proceeds by showing that trajectories of the limiting ordinary differential equation are bounded, that welfare increases over time, and then using Lasalle's invariance theorem to prove that the limiting points of these dynamics coincide with equilibrium points of the ordinary differential equation; showing that the equilibrium points coincide with the maximum of (10) completes the proof.

What makes this result especially useful is that benefit maximization on the part of a user conforms *exactly* to the user trying to maximize its rate through the path reselection process. Thus, this path reselection policy is easy to implement: at random times the session initiates data transfer using the coordinated rate controller over a new set of paths and measures the achieved throughput, dropping either the old path set or new path set depending on which achieves lower throughput. This equivalence is a consequence of the assumption that the utility U is strictly concave and continuously differentiable.

5.2. Uncoordinated congestion control

As one might expect by now, the story is not as clean in the case of uncoordinated control, and no monotonicity result exists. Indeed, for a symmetric triangle network described in Key et al.,[11] with three source-destination session types, allowing each session to use the two-link path as well as the direct path *decreases* throughput. However, random resampling is still beneficial provided that the uncoordinated control exhibits no RTT bias. If a session is given a set of paths to draw from, then the random resampling strategy described earlier maximizes welfare without the need to use all paths. Moreover, it suffices for sessions to use a greedy rate optimization strategy to determine which set of paths to keep in order to ensure welfare maximization. The reader is referred to Key et al.[11] for further details.

6. DISCUSSION AND DEPLOYMENT

Till now, we have focused on networks supporting workloads consisting of *persistent or infinite backlog flows*. Moreover, the emphasis has been on the effect that multipath has on aggregate utility. In this section we consider workloads consisting of finite length flows that arrive randomly to the network. Our metric will be the capacity of the network to handle such flows. We will observe that several results from previous sections have their counterparts when we focus on finite flows.

As before, we represent a network as a capacitated undirected graph $G = (V, E, C)$ supporting a finite set of *flow classes*, $\mathcal{S}$ with attendant sets of paths $\{\mathcal{R}(s)\}$. We assume that class s sessions arrive at rate α_s according to a Poisson process and that they introduce independent and identical exponentially distributed workloads with a mean number of bits $1/\mu_s$. We introduce the notion of a *capacity region* for this network, namely the sets of $\{\alpha_s\}$ and $\{\mu_s\}$ for which there exists some rate allocation over the paths available to the sessions such that the time required for sessions to complete their downloads are finite.

In the case of coordinated control, it is possible to derive the following monotonicity result with respect to the capacity region of the network. Consider a network G that supports a set S of flow classes with arrival rates $\{\alpha_s\}$ and loads $\{\mu_s\}$. Let $\{\mathcal{R}(s)\}$ and $\{\mathcal{R}'(s)\}$ be two collections of paths for these classes that satisfies $\mathcal{R}(s) \subseteq \mathcal{R}'(s)$ for each $s \in \mathcal{S}$ and suppose that each session applies coordinated rate and path control over these paths. Then if $\{\alpha_s\}$, $\{\mu_s\}$, lie within the capacity region of the network with path sets $\{\mathcal{R}(s)\}$, they lie in the capacity region of the network with path sets $\{\mathcal{R}'(s)\}$ as well.

Remark 2. *It is easy to find examples where the capacity region strictly increases with the addition of more paths.*

Remark 3. *Although this result is stated for the case of exponentially distributed workloads, it is straightforward to show that it holds for any workload whose distribution is characterized by a decreasing failure rate. This includes heavy-tailed distributions such as Pareto.*

It is interesting to ask the same question about the capacity region when uncoordinated control is used by all flows. Unfortunately, similar to the infinite session workload case, no such monotonicity property exists.

It is also interesting to ask the question as to which controller yields the larger capacity region. As in the case for finite flows, we can show that for a given network configuration (G, $\mathcal{S}$, and $\mathcal{R}$ fixed), if $\{\alpha_s : s \in \mathcal{S}\}$, $\{\mu_s : s \in \mathcal{S}\}$ lies within the capacity region of the network when operating with an uncoordinated control, then they lie within the capacity region of the network when operating under coordinated control as well.

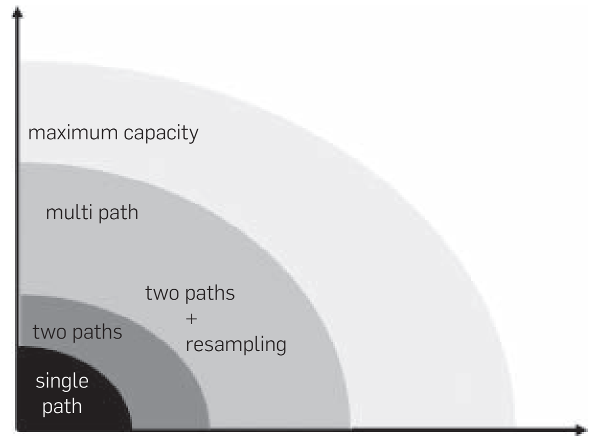
Remark 4. *It is easy to construct cases where the converse is not true. For instance, the symmetric triangle with single and two-link routing mentioned for fixed flows is such an example (see Key and Massoulié[12]).*

We conclude from this monotonicity property for coordinated control that more is better. However, improved capacity comes at the cost of increased complexity at the end-host, namely maintenance of state for each path and executing rate controllers over each path. Fortunately, as in the case of infinite backlogged sessions, this is not necessary. It suffices for a session to maintain a small set of paths, say two paths, and continually try out random paths from the set of paths available to it, and drop the path which provides it with the poorest performance, say throughput. Note the similarity of this process to that of BitTorrent, which periodically drops the connection providing the lowest throughput and replacing it with a random new connection. Interestingly enough, this multipath algorithm coupled with random resampling achieves the same capacity region as one that requires flows to utilize all paths. Indeed, we can prove the analogy of 5.1.

Theorem 5. *Assume that class s sessions use all paths from $\mathcal{R}(s)$. Assume the set of loads $\{\alpha_s\}$ and $\{\mu_s\}$ lies within the network capacity region. Consider an approach where a class s session uses a subset of paths from $\mathcal{R}(s)$, randomly samples a new path set according to a Poisson process with rate γ_s and drops the worst of the two path sets. Then $\{\alpha_k\}$ and $\{\mu_k\}$ also lie within the capacity region when flows use this resampling approach in the limit as $\gamma_s \to \infty$.*

Figure 3 illustrates and summarizes our capacity results. As before it is also interesting to ask about the effect of uncoordinated control coupled with random sampling on capacity. Surprisingly enough, uncoordinated control on a small set of paths coupled with random resampling can often increase capacity over uncoordinated control over the entire set of paths.

Figure 3. Capacity region under multipath with and without resampling.



6.1. Deployment

To effectively deploy multipath, key ingredients are first, diversity, which is achieved through a combination of multihoming and random path sampling, and second, path selection and multipath streaming using a congestion controller that actively streams along the best paths from a working set. Although home-users are currently often limited in their choice of Internet Service Provider (ISP) and hence cannot multihome, in contrast campus or corporate nodes often have diverse connections, via different ISPs or through 3G wireless and wired connectivity. Moreover, the growth of wireless hotspots, wireless mesh and broadband wireless in certain parts of the globe means that even home-users may become multihomed in the future. Recent figures[1] suggest that 60% of stub-ASes (those which do not transit traffic) are multihomed, and de Launois[5] claims that with IPv6 type multihoming there are at least two disjoint paths between such stub-ASes.

The multipath controllers we have outlined need to be put into practice. Some high-level algorithms designs are considered in Kelly and Voice[10] and Han et al.,[7] and practical questions are addressed in Raiciu et al.[18] Translating from algorithms derived from fluid models to practical packet-based implementations does require care; however, we believe this to be perfectly feasible in practice. Indeed, the IETF has a current Multipath TCP working group, which is looking into adding multipath into TCP.

7. SUMMARY

There are potentially significant gains from combining multipath routing with congestion control. Two different flavors of control are possible: one which coordinates transfers across the multiple paths; and another uncoordinated control with sets up parallel connections. The uncoordinated approach is simpler to implement; however, it can suffer from poorer performance while coordinated control is better performing and intrinsically "fairer." We have contrasted the two types of control, and shown that with fixed path choices uncoordinated control can yield inferior performance, halving throughput in one example.

If path-choices are allowed to be chosen optimally or "selfishly" by the end-system, then coordinated control reaches the best systemwide optimum; as indeed does uncoordinated control, but only if the control objective is the same for all paths (unlike current TCP), and also only if all users agree to use the same number of parallel paths (connections). This optimum can also be reached by limiting each session to a small number of path choices (e.g., 2) but allowing paths to be resampled and better paths to replace existing ones.

This suggests that good design choices for multipath controllers are coordinated controllers or uncoordinated controllers with the RTT bias removed.

Acknowledgment

This work was supported in part by the NSF under award CNS-0519922.

References

1. Agarwal, S., Chuah, C.-N., Katz, R. OPCA: Robust interdomain policy routing and traffic control. In *Proceedings of the IEEE Openarch* (April 2003).
2. Andersen, D., Balakrishnan, H., Kaashoek, F., Rao, R. Improving Web availability for clients with MONET. In *Proceedings of the NSDI 2005* (July 2005).
3. Bertsekas, D., Gallager, R. *Data Networks*. Longman Higher Education, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1992.
4. Cohen, B. Incentives build robustness in BitTorrent. In *Proceeding of P2P Economics workshop* (June 2003).
5. de Launois, C., Quoitin, B. Bonaventure, O. Leveraging network performance with IPv6 multihoming and multiple provider-dependent aggregatable prefixes. *Comput. Netw. 50*, 8 (2006), 1145–1157.
6. Gummadi, K., Madhyastha, H., Gribble, S., Levy, H., Wetherall, D. Improving the reliability of Internet paths with one-hop source routing. In *Proceedings of the 6th OSDI* (Dec. 2004).
7. Han, H., Shakkottai, S., Hollot, C., Srikant, R., Towsley, D. Multi-path TCP: a joint congestion control and routing scheme to exploit path diversity in the Internet. *IEEE/ACM Trans. Netw. 14*, 6 (Dec. 2006), 1260–1271.
8. Kelly, F., Maulloo, A., Tan, D. Communication networks: shadow prices, proportional fairness and stability. *J. Oper. Res. Soc. 49*, (1998), 237–252.
9. Kelly, F.P. Mathematical modelling of the Internet. *Mathematics Unlimited–2001 and Beyond*. B. Engquist and W. Schmid, eds. Springer-Verlag, New York, 2001, 685–702.
10. Kelly, F.P., Voice, T. Stability of end-to-end algorithms for joint routing and rate control. *ACM SIGCOMM Comput. Comm. Rev. 35*, 2 (2005), 5–12.
11. Key, P., Massoulié, L., Towsley, D. Path selection and multipath congestion control. In *INFOCOM07* (May 2007).
12. Key, P., Massoulié, L. Fluid models of integrated traffic and multipath routing. *Queueing Syst. 53*, 1 (June 2006), 85–98.
13. Key, P., Massoulié, L., Towsley, D. Multipath routing, congestion control and load balancing. In *ICASSP 2007* (Apr. 2007).
14. Kodialam, M., Lakshman, T., Sengupta, S. Efficient and robust routing of highly variable traffic. In *HotNets* (2004).
15. Kunniyur, S., Srikant, R. End-to-end congestion control schemes: utility functions, random losses and ECN marks. In *INFOCOM 2000* (2000).
16. Mitzenmacher, M., Richa, A. Sitaraman, R. The power of two random choices: a survey of the techniques and results. *Handbook of Randomized Computing*. P. Pardalos, S. Rajasekaran, and J. Rolim, eds. Kluwer Academic Publishers, Dordrecht, 2001, 255–312.
17. Mo, J., Walrand, J. Fair end-to-end window based congestion control. In *SPIE 98, International Symposium on Voice, Video and Data Communications* (1998).
18. Raiciu, C., Wischik, D., Handley, M. Practical congestion control for multipath transport protocols. UCL Technical Report (2010).
19. Ramakrishnan, K., Floyd, S., Black, D. The addition of explicit congestion notification (ECN) to IP. Technical Report RFC3168, IETF (Sept. 2001).
20. Srikant, R. *The Mathematics of Internet Congestion Control*. Birkhauser, Boston, 2003.
21. Zhang-Shen, R., McKeown, N. Designing a predictable Internet backbone network with Valiant load-balancing. In *IWQoS* (June 2005).

Peter Key (peter.key@microsoft.com), Microsoft Research, Cambridge, UK.

Laurent Massoulié (laurent.massoulie@technicolor.com), Thomson Technology Paris Laboratory, 1, Issy-les-Moulineaux-Moulineau, France.

Don Towsley (towsley@cs.umass.edu), Department of Computer Science, University of Massachusetts, Amherst, MA.

Air Force Institute of Technology (AFIT) Dayton, Ohio
Department of Electrical and Computer Engineering
Graduate School of Engineering and Management
Faculty Positions in Computer Science or Computer Engineering

The Department of Electrical and Computer Engineering is seeking applicants for tenure track positions in computer science or computer engineering. The department is particularly interested in receiving applications from individuals with strong backgrounds in formal methods (with emphasis on cryptography), software engineering, bioinformatics, computer architecture/VLSI systems, and computer networks and security. The positions are at the assistant professor level, although qualified candidates will be considered at all levels. Applicants must have an earned doctorate in computer science or computer engineering or closely related field and must be U.S. citizens. These positions require teaching at the graduate level as well as establishing and sustaining a strong research program.

AFIT is the premier institution for defense-related graduate education in science, engineering, advanced technology, and management for the U.S. Air Force and the Department of Defense (DoD). Full details on these positions, the department, and application procedures can be found at: http://www.afit.edu/en/eng/employment_faculty.cfm

Review of applications will begin immediately and will continue until the positions are filled. The United States Air Force is an equal opportunity, affirmative action employer.

California State University, Fullerton
Assistant Professor

The Department of Computer Science invites applications for a tenure-track position at the **Assistant Professor** level starting fall 2011. For a complete description of the department, the position, desired specialization and other qualifications, please visit http://diversity.fullerton.edu/.

Cal Poly Pomona
Assistant Professor

The Computer Science Department invites applications for a tenure-track position at the rank of Assistant Professor to begin Fall 2011. We are particularly interested in candidates with specialization in Software Engineering, although candidates in all areas of Computer Science will be considered, and are encouraged to apply. Cal Poly Pomona is 30 miles east of L.A. and is one of 23 campuses in the California State University. The department offers an ABET-accredited B.S. program and an M.S. program. Qualifications: Possess, or complete by September 2011, a Ph.D. in Computer Science or closely related area. Demonstrate strong English communication skills, a commitment to actively engage in the teaching, research and curricular development activities of the department at both undergraduate and graduate levels, and ability to work with a diverse student body and multicultural constituencies. Ability to teach a broad range of courses, and to articulate complex subject matter to students at all educational levels. First consideration will be given to completed applications received no later than December 15, 2010. Contact: Faculty Search Committee, Computer Science Department, Cal Poly Pomona, Pomona, CA 91768. Email: cs@csupomona.edu. Cal Poly Pomona is an Equal Opportunity, Affirmative Action Employer. Position announcement available at: http://academic.csupomona.edu/faculty/positions.aspx. Lawful authorization to work in US required for hiring.

Carnegie Mellon University
School of Design
IxD Faculty Position

School of Design at Carnegie Mellon University
IxD Faculty Position
Application deadline December 3, 2010
Submit application to johnz@cs.cmu.edu.
View complete job description at http://bit.ly/cGrgeC.

Columbia University
Department of Computer Science
Tenured or Tenure-Track Faculty Positions

The **Department of Computer Science at Columbia University** in New York City invites applications for tenured or tenure-track faculty positions. The search committee is especially interested in candidates who through their research, teaching, and/or service will contribute to the diversity and excellence of the academic community. Appointments at all levels, including assistant professor, associate professor and full professor, will be considered. Priority themes for the department include Computer Systems, Software, Artificial Intelligence, Theory, and Computational Biology. Candidates who work in specific technical areas including, but not limited to, Computer Graphics, Human-Computer Interaction, Simulation, and Animation, with research programs that can significantly impact the above priority themes are particularly welcome to apply. Candidates doing research at the interface of computer sciences and the life sciences and the physical sciences are also encouraged to apply.

Assistant Professors at Columbia are academic officers holding the doctorate or its professional equivalent who are beginning a career of independent scholarly research and teaching.

Associate Professors at Columbia are academic officers holding the doctorate or its professional equivalent who have demonstrated scholarly and teaching ability and show great promise of attaining distinction in their fields of specialization.

Professors at Columbia are academic officers holding the doctorate or its professional equivalent who are widely recognized for their distinction. Candidates for senior-level appointment must have a distinguished record of achievement and evidenced by leadership in their field of expertise, publications, professional recognition, as well as a commitment to excellence in teaching.

Candidates must have a Ph.D. degree, DES, or equivalent degree by the starting date of the appointment and are expected to establish a strong research program and excel in teaching both undergraduate and graduate courses.

Our department of 36 tenure-track faculty and 1 lecturer attracts excellent Ph.D. students, virtually all of whom are fully supported by research grants. The department has active ties with major industry partners including Adobe, Autodesk, Canon, Disney, Dreamworks, Microsoft, Nvidia, Google, Sony, Weta, Yahoo! and also to the nearby research laboratories of AT&T, Google, IBM (T.J. Watson), NEC, Siemens, Telcordia Technologies and Verizon. Columbia University is one of the leading research universities in the United States, and New York City is one of the cultural, financial, and communications capitals of the world. Columbia's tree-lined campus is located in Morningside Heights on the Upper West Side.

Applicants should apply online at:
academicjobs.columbia.edu/applicants/Central?quickFind=54003

and should submit electronically the following: curriculum-vitae including a publication list, a statement of research interests and plans, a statement of teaching interests, names with contact information of three references, and up to four pre/reprints. Applicants can consult www.cs.columbia.edu for more information about the department.

The position will close no sooner than December 31, 2010, and will remain open until filled. Columbia University is an Equal Opportunity/Affirmative Action employer

DePaul University
Assistant/Associate Professor

The School of Computing at DePaul University invites applications for a tenure-track position in distributed systems. We seek candidates with a research interest in data-intensive distributed systems, cloud computing, distributed databases, or closely related areas. For more information, see https://facultyopportunities.depaul.edu/applicants/Central?quickFind=50738.

Eastern New Mexico University
Instructor of Computer Science

For more information visit www.enmu.edu/services/hr or call (575) 562-2115. All employees must pass a pre-employment background check. AA/EO/Title IX Employer

Eastern Washington University
Tenure-track Position

The Computer Science Department at Eastern Washington University invites applications for a tenure-track position starting Sept 2011. Please visit: http://access.ewu.edu/HRRR/Jobs.xml for complete information. For questions contact Margo Stanzak (509) 359-4734

Goucher College
Visiting Assistant Professor, Computer Science

Applications are invited for a three year visiting assistant professor position beginning August 2011. This is a one-year appointment, renewable for up to two additional years. A Ph.D. in computer science, or a closely related field, is preferred (non-Ph.D. applicants must be ABD).

Applicants should have experience teaching a wide range of courses at all levels of the computer science curriculum. Preference will be given to applicants with a systems background, but applicants from all areas of computer science will be considered. Applicants are expected to present strong teaching credentials. Women and minorities are encouraged to apply.

Application review will begin January 18, 2011 and continue until the position is filled.

Interested applicants must apply online:
http://goucher.interviewexchange.com/candapply.jsp?JOBID=21846

Please submit the following application materials online:

- Curriculum Vitae
- Cover letter
- A personal statement describing your interest in teaching at a small liberal arts college

Three letters of recommendation and official graduate transcripts should be forwarded separately to: Human Resources, Goucher College, 1021 Dulaney Valley Road, Baltimore, MD 21204-2794.

Goucher College is an Equal Opportunity Employer.

Hitachi Research
Research Scientist
Storage Architecture

Hitachi San Jose Research Center is a premier research center with more than 100 scientists working in many exciting fields including storage architecture, consumer electronics, storage technology and nanotechnology. The job opening is in the area of research of storage architecture and systems, more specifically operating systems, novel file systems, reliability of storage devices and systems and novel storage and computing architectures. Ideal candidate would work both independently and as a part of a storage architecture team in conceiving, prototyping and guiding development of new storage related projects and ideas, as well as writing invention disclosures, academic papers and publications and participating in scientific societies and industry associations. Postdoctoral candidates are welcome to apply and propose more specific research programs.

Job Requirements
PhD in Computer Science with a proven publication track record and implementation experience in system architecture, operating systems, file systems, and embedded systems.

Send applications (a full Curriculum Vitae and short description of research interests) to Zvonimir.Bandic@hitachigst.com

Illinois Institute of Technology
Department of Computer Science

Applications are invited for a tenure-track assistant professor position in Computer Science beginning Fall 2011. Excellence in research, teaching and obtaining external funding is expected. While strong candidates from all areas of computer science will be considered, applicants from general data areas such as database, data mining, information security, information retrieval, and data understanding and processing are especially encouraged.

The Department offers B.S., M.S., and Ph.D. degrees in Computer Science and has research

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Faculty Positions in Computer Science
at Ecole polytechnique fédérale de Lausanne (EPFL)

The School of Computer and Communication Sciences at EPFL invites applications for faculty positions in computer science. We are primarily seeking candidates for **tenure-track assistant professor positions, but suitably qualified candidates for senior positions will also be considered.**

Successful candidates will develop an independent and creative research program, participate in both undergraduate and graduate teaching, and supervise PhD students.

Candidates from all areas of computer science will be considered, but preference will be given to candidates with interests in **algorithms, bio-informatics and verification.**

EPFL offers internationally competitive salaries, significant start-up resources, and outstanding research infrastructure.

To apply, please follow the application procedure at http://icrecruiting.epfl.ch. The following documents are requested in PDF format: curriculum vitae, including publication list, brief statements of research and teaching interests, names and addresses (including e-mail) of 3 references for junior positions, and 6 for senior positions. Screening will start on **January 1, 2011**. Further questions can be addressed to :

Professor Willy Zwaenepoel, Dean
School of Computer and Communication Sciences, EPFL
CH-1015 Lausanne, Switzerland
recruiting.ic@epfl.ch

For additional information on EPFL, please consult: http://www.epfl.ch or http://ic.epfl.ch

EPFL is an equal opportunity employer.

CALIFORNIA STATE UNIVERSITY, FRESNO
Computational Computer Scientist

Assistant Professor of Computer Science (#11583)
Position will teach computational science and computer science, help organize, utilize, and apply high-performance computing clusters with a proposed Computational Science Center. This position is part of a University-wide cohort of new faculty with a broad emphasis in the area of Urban and Regional Transformation. An earned doctorate (Ph.D) in Computer Science or closely related field from an accredited institution is required for appointment to a tenure-track position. Preference will be given to candidates with post-doctoral research or industrial experience. Visit jobs.csufresno.edu for full announcement and application. Application materials should be submitted online by 1/12/2011. California State University, Fresno is an affirmative action/equal opportunity institution.

Assistant Professor of Computer Science (#11589)
Position will teach undergraduate and graduate computer science to majors and non-majors. An earned doctorate (Ph.D) in Computer Science or closely related field is required for appointment to a tenure-track position. Preference will be given to candidates with post-doctoral research or industrial experience, and whose specialties are in software engineering, computer/operating systems, database/information systems, graphics/visual computing, intelligent systems, or algorithms. Visit jobs.csufresno.edu for full announcement and application. Application materials should be submitted online by 1/12/2011. California State University, Fresno is an affirmative action/equal opportunity institution.

100 graduate students, and 250 undergraduate students and offers BS, MS, MSE, and PhD degrees. Computing facilities include a large network of servers, workstations and PCs with more than 300 machines and a Beowulf cluster with 1000+ processors. The department building has a wireless network and state-of-the-art media-equipped classrooms. The department hosts several laboratories for embedded systems, software analysis, robotics, computational engineering and science, and data-mining. Details of the CIS Department can be found at the URL http://www.cis.ksu.edu/.

Applicants must be committed to both teaching and research, and have an excellent research and teaching track record. Applicants should have a PhD degree in computer science or related disciplines; salary will be commensurate with qualifications. Applications must include descriptions of teaching and research interests along with copies of representative publications.

Preference will be given to candidates who will compliment the existing areas of strengths of the department which include high assurance systems, tools for developing, testing and verifying software systems, static analysis, model-driven computing, programming languages, security, and medical device software.

Please send applications to Chair of the Recruiting Committee, Department of Computing and Information Sciences, 234 Nichols Hall, Kansas State University, Manhattan, KS 66506 (email: Recruiting@cis.ksu.edu). Review of applications will commence January 3rd, 2011 and continue until the position is filled.

Kansas State University is an Equal Opportunity Employer and actively seeks diversity among its employees. Background checks are required.

Lingnan University
Chair Professor / Professor

The Department of Computing and Decision Sciences at the Lingnan University is seeking a Chair Professor/Professor with outstanding teaching and research experience in one or more of the following areas: Information Systems, Operations Management, Management Science and Statistics. Please visit http://www.ln.edu.hk/job-vacancies/acad/10-170 for details and quote post ref: 10/170/CACM in Form R1.

Loyola University Maryland
Assistant Professor, Computer Science

Loyola University Maryland invites applications for the position of Clare Boothe Luce Professor in the Department of Computer Science, with an expected start date of fall 2011 at the level of Assistant Professor. We are seeking an enthusiastic individual committed to excellent teaching and a continuing, productive research program. A Ph.D. in Computer Science, Computer Engineering, or a closely related field is required. Candidates in all areas of specialization will be considered. The position is restricted by the Clare Boothe Luce bequest to the Henry Luce Foundation to women who are U.S. citizens. Duties of the position include teaching undergraduate and professional graduate computer science courses, conducting research in computer science to establish a record of scholarly, peer-reviewed work, and providing service to the department and the University. The position comes with dedicated funding for course releases as well as a professional development fund. The course releases will mean a reduced load in the 1st-3rd and 5th years, and a guaranteed early research sabbatical in the 4th year after successful midterm review. More information is available at www.loyola.edu/cs and www.loyola.edu Applicants must submit the following online (http://careers.loyola.edu): a letter of application listing teaching and research interests, a curriculum vitae, and contact information for three references. For full consideration applications must be received by January 31, 2011. Apply URL: https://careers.loyola.edu/applicants/Central?quickFind=52395

Max Planck Institute for Software Systems (MPI-SWS)
Tenure-track openings

Applications are invited for tenure-track and tenured faculty positions in all areas related to the study, design, and engineering of software systems. These areas include, but are not limited to, data and information management, programming systems, software verification, parallel, distributed and networked systems, and embedded systems, as well as cross-cutting areas like security, machine learning, usability, and social aspects of software systems. A doctoral degree in computer science or related areas and an outstanding research record are required. Successful candidates are expected to build a team and pursue a highly visible research agenda, both independently and in collaboration with other groups. Senior candidates must have demonstrated leadership abilities and recognized international stature.

MPI-SWS, founded in 2005, is part of a network of eighty Max Planck Institutes, Germany's premier basic research facilities. MPIs have an established record of world-class, foundational research in the fields of medicine, biology, chemistry, physics, technology and humanities. Since 1948, MPI researchers have won 17 Nobel prizes. MPI-SWS aspires to meet the highest standards of excellence and international recognition with its research in software systems.

To this end, the institute offers a unique environment that combines the best aspects of a university department and a research laboratory:

a) Faculty receive generous base funding to build and lead a team of graduate students and post-docs. They have full academic freedom and publish their research results freely.

b) Faculty supervise doctoral theses, and have the opportunity to teach graduate and undergraduate courses.

c) Faculty are provided with outstanding technical and administrative support facilities as well as internationally competitive compensation packages.

MPI-SWS currently has 8 tenured and tenure-track faculty, and is funded to support 17 faculty and about 100 doctoral and post-doctoral positions. Additional growth through outside funding is possible. We maintain an open, international and diverse work environment and seek applications from outstanding researchers regardless of national origin or citizenship. The working language is English; knowledge of the German language is not required for a successful career at the institute.

The institute is located in Kaiserslautern and Saarbruecken, in the tri-border area of Germany, France and Luxembourg. The area offers a high standard of living, beautiful surroundings and easy access to major metropolitan areas in the center of Europe, as well as a stimulating, competitive and collaborative work environment. In immediate proximity are the MPI for Informatics, Saarland University, the Technical University of Kaiserslautern, the German Center for Artificial Intelligence (DFKI), and the Fraunhofer Institutes for Experimental Software Engineering and for Industrial Mathematics.

Qualified candidates should apply online at http://www.mpi-sws.org/application. The review of applications will begin on January 3, 2011, and applicants are strongly encouraged to apply by that date; however, applications will continue to be accepted through January 2011.

The institute is committed to increasing the representation of minorities, women and individuals with physical disabilities in Computer Science. We particularly encourage such individuals to apply.

Mississippi State University
Head
Department of Computer Science and Engineering

Applications and nominations are being sought for the Head of the Department of Computer Science and Engineering (www.cse.msstate.edu) at Mississippi State University. This is a 12-month tenure-track position.

The successful Head will provide:

- Vision and leadership for nationally recognized computing education and research programs
- Exceptional academic and administrative skills
- A strong commitment to faculty recruitment and development

Applicants must have a Ph.D. in computer science, software engineering, computer engineering, or a closely related field. The successful candidate must have earned national recognition by a distinguished record of accomplishments in computer science education and research. Demonstrated administrative experience is desired, as is teaching experience at both the undergraduate and graduate levels. The successful candidate must qualify for the rank of professor.

Please provide a letter of application outlining your experience and vision for this position, a curriculum vita, and names and contact information of at least three professional references. Application materials should be submitted online at http://www.jobs.msstate.edu/.

Screening of candidates will begin February 15, 2011 and will continue until the position is filled. Mississippi State University is an AA/EOE institution. Qualified minorities, women, and people with disabilities are encouraged to apply.

Please direct any questions to Dr. Nicolas Younan, Search Committee Chair (662-325-3912 or younan@ece.msstate.edu).

National Taiwan University
Professor-Associate Professor-Assistant Professor

The Department of Computer Science and Information Engineering has faculty openings at all ranks beginning in August 2011. Highly qualified candidates in all areas of computer science/engineering are invited to apply. A Ph.D. or its equivalent is required. Applicants are expected to conduct outstanding research and be committed to teaching. Candidates should send a curriculum vitae, three letters of reference, and supporting materials before February 28, 2011, to Prof Kun-Mao Chao, Department of Computer Science and Information Engineering, National Taiwan University, No 1, Sec 4, Roosevelt Rd., Taipei 106, Taiwan.

NEC Laboratories America, Inc
Research Staff Positions

NEC Laboratories America, Inc. is a vibrant industrial research center, conducting research in support of NEC's U.S. and global businesses. Our research program covers many areas, reflecting the breadth of NEC business, and maintains a balanced mix of fundamental and applied research. We have openings in the following research areas:

Research Staff Members
The **Large-Scale Distributed Systems** group conducts advanced research in the area of design, analysis, modeling and evaluation of distributed systems. Our current focus is to create innovative technologies to build next generation large-scale computing platforms, and to simplify and automate the management of complex IT systems and services. The group is seeking research staff members in the area of distributed systems and networks. The candidates must have a PhD in CS/CE with strong publication records on the following topics:
- distributed systems and networks
- operating systems and middleware
- performance, reliability, dependability and security
- data centers and cloud computing
- virtualization and system management
- system modeling and statistical analysis

The **Computing Systems Architecture** department seeks to innovate, design, evaluate, and deliver parallel systems for high-performance, energy-efficient enterprise computing. The group is seeking senior and junior level research staff as follows. Candidates for Research Staff Member must have a PhD in CS/CE or EE with strong research record and excellent credentials in the international research community. Applicants must demonstrate competency in at least one of the following areas:
- heterogeneous cluster architectures
- parallel programming models and runtimes
- key technologies to accelerate performance and low power consumption of enterprise applications on heterogeneous clusters

The Storage Systems department engages in research in all aspects of storage systems with an emphasis on large scale reliable distributed systems. The group is seeking candidates with an interest in file and storage systems, cloud computing, or related areas. Applicants must have a PhD in CS and a strong publication record in the above topics. Required skills are:
- proactive and assume leadership in proposing and executing innovative research projects
- develop advanced prototypes leading to demonstration in industrial environment
- initiate and maintain collaborations with academic and industrial research communities

Postdoctoral Researchers
The **Machine Learnin**g group conducts research on various aspects of machine intelligence, from the exploration of new algorithms to applications in data mining and semantic comprehension. Ongoing projects focus on text and video analysis, digital pathology, and bioinformatics. The group is seeking postdoctoral researchers with PhD in CS and experience in bioinformatics (emphasis on genomics or proteomics a plus) or text analysis and/or text mining. Required skills and experience are:
- Strong publication record in top machine learning, data mining or related conferences and journals
- Solid knowledge in math, optimization, and statistical inference
- Hands-on experiences in implementing large-scale learning algorithms and systems
- Good problem solving skills, with strong software knowledge

Associate Research Staff Members
Candidates for Associate Research Staff Member in the **Computing Systems Architecture** department must have an MS in CS/CE or EE with strong motivation and skill set to prototype/transfer innovate research results into industry practice. Expertise in at least one of the above parallel computing areas is desirable.

The **Storage Systems** department is seeking applicants for an Associate Research Staff Member. The successful candidate will have an MS in CS or equivalent and the following skills:
- Solid understanding of operating systems
- Experience in systems programming under Linux/Unix
- Experience with performance evaluation and tuning
- Strong algorithms, data structures and multithreaded programming experience
- Good knowledge of C++ and OOD/OOP
- Proactive with can-do attitude and work well in small teams

For more information about NEC Labs and these openings, access http://www.nec-labs.com and submit your CV and research statement through our career center.

EOE/AA/MFDV

Northeastern University
Boston, Massachusetts
College of Computer and Information Science

We Invite applications for tenure-track faculty positions in computer science and information science, beginning in Fall 2011. Applicants at all ranks will be considered. A PhD in computer science, information science or a related field is required.

Candidates will be considered from all major disciplines in computer and information science. We particularly welcome candidates who can contribute to our strong research groups in software reliability (formal methods, programming languages, software engineering) and systems and networks.

The College maintains a strong research program with significant funding from the major federal research agencies and private industry. The College has a diverse full-time faculty of 30. Four faculty members have joint appointments with other disciplines, specifically, electrical and computer engineering, health sciences, physics and political science, and contribute to interdisciplinary initiatives in information assurance, network science and health informatics. The College has approximately 520 undergraduates, 350 Masters, and 65 Ph.D. students.

Northeastern University has made major investments over the course of the last several years in the broad areas of Health, Security and Sustainability. The College has been a major participant in the recruitment of faculty who can contribute to these themes and will continue to do so this year as well with an additional three interdisciplinary searches ongoing in Health Informatics, Information Assurance and Game Design and Interactive Media.

Northeastern University is located on the Avenue of the Arts in Boston's historic Back Bay. The College occupies a state of the art building opposite Boston's Museum of Fine Arts.

Additional information and instructions for submitting application materials may be found at the following web site: http://www.ccs.neu.edu/. Screening of applications begins immediately and will continue until the search is completed.

Northeastern University is an Equal Opportunity/Affirmative Action Employer. We strongly encourage applications from women and minorities.

Northeastern University
Open Rank - Interdisciplinary

Northeastern University is seeking a faculty member at an open rank for an interdisciplinary appointment in the College of Computer and Information Science and the College of Arts, Media and Design to start in the Fall of 2011.

The successful candidate will contribute to shaping the research, academic, and development goals of the cross-disciplinary areas of Game Design and Interactive Media at both the undergraduate and the graduate levels.

It is expected that the candidate for this position will possess an excellent track record in research/scholarship, publication, grant acquisition, and teaching. A terminal degree, either PhD or MFA depending on the candidate's field, is required.

Contact: Terrence Masson - Email: t.masson@neu.edu

Northeastern University, Boston, Massachusetts
Full or Associate Professor - Health Informatics and Interfaces

The College of Computer and Information Science and the Bouvé College of Health Sciences invite applications for a faculty position in Health

Informatics. A Ph.D. level degree in Health or Medical Informatics, Computer Science, Information Science, or a health-related discipline, together with a proven ability to secure grant funding for research using advanced technology in the health domain, is required.

Building upon our successful joint Master of Science degree program in Health Informatics, and our many graduate and undergraduate degree programs in health sciences, nursing, pharmacy, computer science, and information science, we are interested in growing our faculty in the general area of health interfaces, which includes technologies that patients interact with directly, health informatics, and technology design for health and wellness systems. The candidate would play a key role in launching a new interdisciplinary Ph.D.-level degree program in this area. Faculty in our colleges are currently working on multiple NIH-funded research projects in consumer informatics, clinical informatics, behavioral informatics, and assistive technologies, and we are particularly interested in faculty candidates who can expand or complement our work in these areas. Topics of interest include the use of mobile technologies to monitor and manage health, the use of virtual agents for physical exercise and health management, the development of assistive communication aids, the use of artificial intelligence to study mental and physical health behavior, and the development and evaluation of other novel technologies to study health behavior and improve health outcomes. We are interested in candidates who create new tools and candidates who specialize in evaluation of new technologies in field research. Northeastern University is making a major investment in interdisciplinary health research, with several recent hires and additional open interdisciplinary faculty searches in Health Systems, Health Policy, Urban Environment and Health and Administration.

Additional Information
Recognizing the importance of multidisciplinary approaches to solving complex problems facing society, Northeastern is hiring faculty in several areas related to this search. Searches are currently underway in health care policy/ management, health systems engineering, health law, and urban health. We will consider hiring a multidisciplinary group as a 'cluster hire'. Candidates may choose to form a team and propose an innovative and translational research and educational direction and apply to more than one of the position announcements. Information on these positions and on cluster applications can be obtained from the http://www.northeastern.edu/hrm/ web site.

Equal Employment Opportunity
Northeastern University is an Equal Opportunity, Affirmative Action Educational Institution and Employer, Title IX University. Northeastern University strongly encourages applications from minorities, women and persons with disabilities.

How To Apply
Applicants should submit a letter of interest, curriculum vitae, and the contact information of at least five references. Submission is online via http://www.ccs.neu.edu/. Screening of applications begins November 30, 2010 and will continue until the position is filled.

Please direct inquiries to Professor Stephen Intille (S.Intille@neu.edu).

Oregon State University
School of Electrical Engineering and Computer Science
Two tenure-track Professorial positions in Computer Science

The School of Electrical Engineering and Computer Science at Oregon State University invites applications for two tenure-track professorial positions in Computer Science. Exceptionally strong candidates in all areas of Computer Science are encouraged to apply. We are building research and teaching strengths in the areas of open source software, internet and social computing, and cyber security, so our primary need is for candidates specializing in software engineering, database systems, web/distributed systems, programming languages, and HCI. Applicants should demonstrate a strong commitment to collaboration with other research groups in the School of EECS, with other departments at Oregon State University, and with other universities.

The School of EECS supports a culture of energetic collaboration and faculty are committed to quality in both education and research. With 40 tenure/tenure-track faculty, we enroll 160 PhD, 120 MS and 1200 undergraduate students. OSU is the only Oregon institution recognized for its "very high research activity" (RU/VH) by the Carnegie Foundation for the Advancement of Teaching. The School of EECS is housed in the Kelley Engineering Center, a green building designed to support collaboration among faculty and students across campus. Oregon State University is located in Corvallis, a college town renowned for its high quality of life.

For more information, including full position announcement and instructions for application, visit: http://eecs.oregonstate.edu/faculty/openings.php.

OSU is an AAEOE.

Pacific Lutheran University
Assistant Professor

Assistant Professor in the Computer Science and Computer Enginieering Department beginning September 2011. Review of applications will begin February 14, 2011, and continue until the position is filled.

A master's degree is required and a Doctorate is required for tenure. Preferred candidates will have a Ph.D. in Computer Engineering, Computer Science, or a related field; promise of teaching excellence is essential.

Application details and further information about PLU and the CSCE department can be found at www.plu.edu and www.cs.plu.edu. Inquiries may be sent by e-mail to csce@plu.edu. AA/EOE

Penn State Harrisburg
Assistant Professor, Computer Science

Penn State Harrisburg, School of Science, Engineering and Technology, invites applications for the tenure-track position of Assistant Professor of Computer Science effective Fall Semester 2011. The successful candidate will have experience and research interest in software engineering/software design, compilers, or programming languages. Candidates will be evaluated on teaching and research potential. Ph.D. in Computer Science is required. Faculty are expected to teach courses for the B.S. and M.S. degrees in Computer Science, pursue scholarly research and publications, contribute to curriculum development, participate in University and professional service activities, advise undergraduate and graduate students, and serve on graduate level degree committees. For information on Penn State Harrisburg, please visit our websites at www.hbg.psu.edu and www.cs.hbg.psu.edu.

Applicants are invited to submit current curriculum vitae, a list of three references with one reference addressing candidate's teaching effectiveness, a personal statement of research and teaching objectives that includes a list of preferred courses to teach. Please submit credentials to: Chair, Computer Science Search Committee, c/o Mrs. Dorothy J. Guy, Director of Human Resources, Penn State Harrisburg, Box: ACM-33389, 777 West Harrisburg Pike, Middletown, PA 17057-4898.

Application review will begin immediately and continue until the position is filled. Penn State is committed to affirmative action, equal opportunity, and the diversity of its workforce.

Princeton University
Computer Science
Assistant Professor
Tenure-Track Positions

The Department of Computer Science at Princeton University invites applications for faculty positions at the Assistant Professor level. We are accepting applications in all areas of Computer Science.

Applicants must demonstrate superior research and scholarship potential as well as teaching ability. A PhD in Computer Science or a related area is required.

Successful candidates are expected to pursue an active research program and to contribute significantly to the teaching programs of the department. Applicants should include a resume contact information for at least three people who can comment on the applicant's professional qualifications.

There is no deadline, but review of applications will start in December 2010; the review of applicants in the field of theoretical computer science will begin as early as October 2010.

Princeton University is an equal opportunity employer and complies with applicable EEO and affirmative action regulations You may apply online at:

http://www.cs.princeton.edu/jobs Requisition Number: 1000520

Princeton University
Computer Science Department
Postdoc Research Associate

The Department of Computer Science at Princeton University is seeking applications for postdoctoral or more senior research positions in theoretical computer science. Candidates will be affiliated with the Center for Computational Intractability (CCI) or the Princeton Center for Theoretical Computer Science. Candidates should have a PhD in computer Science, a related field, or on track to finish by August 2011. Candidates affiliated with the CCI will have visiting privileges at partner institutions NYU, Rutgers University, and The Institute for Advanced Study. Review of candidates will begin Jan 1, 2011, and will continue until positions are filled. Applicants should submit a CV, research statement, and contact information for three references.

Princeton University is an equal opportunity employer and complies with applicable EEO and affirmative action regulations.

Apply to:http://jobs.princeton.edu/ requisition# 1000829

Princeton University
Visiting Fellows

The Center for Information Technology Policy (CITP) seeks candidates for positions as visiting faculty members or researchers, or postdoctoral research associates for one year appointments the 2011-2012 academic year. Please see our website for additional information and requirements at http://citp.princeton.edu/call-for-visitors/.

If you are interested, please submit a CV and cover letter, stating background, intended research, and salary requirements, to jobs.princeton.edu/applicants/Central?quickFind=60250.

Princeton University is an equal opportunity employer and complies with applicable EEO and affirmative action regulations.

Purdue University
Department of Computer Science
Assistant Professor

The Department of Computer Science at Purdue University invites applications for tenure-track positions at the assistant professor level beginning August 2011. Outstanding candidates in all areas of Computer Science will be considered. Specific needs that have been identified include theory and software engineering. Candidates with a multi-disciplinary focus are encouraged to apply.

The Department of Computer Science offers a stimulating and nurturing academic environment. Forty-four faculty members direct research programs in analysis of algorithms, bioinformatics, databases, distributed and parallel computing, graphics and visualization, information security, machine learning, networking, programming languages and compilers, scientific computing, and software engineering. Information about the department and a detailed description of the open position are available at http://www.cs.purdue.edu.

All applicants should hold a PhD in Computer Science, or a closely related discipline, be committed to excellence in teaching, and have demonstrated potential for excellence in research. The successful candidate will be expected to teach courses in computer science, conduct research in field of expertise and participate in other department and university activities. Salary and benefits are highly competitive. Applicants are strongly encouraged to apply online at https://hiring.science.purdue.edu. Hard copy applications can be sent to: Faculty Search Chair, Department of Computer Science, 305 N. University Street, Purdue University, West Lafayette, IN 47907. Review of applications will begin on November 10, 2010, and will continue until the positions are filled. Purdue University is an Equal Opportunity/Equal Access/Affirmative Action employer fully committed to achieving a diverse workforce.

Purdue University
Computer and Information Technology
Department Head

Computer and Information Technology, Purdue University, Department Head; The College of Technology invites nominations and applications for the position of Department Head, Computer and Information Technology (CIT). The department head reports to the Dean of the College of Technology and is responsible for the strategic leadership of the department in academic, administrative, budgetary, and personnel decisions. Candidates must hold a terminal degree in a related discipline and have the credentials appropriate for appointment at the rank of full professor with tenure at Purdue University. Candidates will share the vision of the University and the department, and have demonstrated strategic and transformative leadership with a proven record of scholarship, external funding, and teaching. A full description of the position and application process is available on line at www.tech.purdue.edu/cit/aboutus/positions.cfm. Purdue University is an equal opportunity, equal access, affirmative action employer.

Rutgers, The State University of New Jersey
Department of Management Science and Information Systems

The Department of Management Science and Information Systems (MSIS) has a tenure-track opening starting Fall 2011 at either the Assistant or Associate Professor level. Candidates should have a Ph.D. in information technology or a related area. A candidate must be an active researcher and have a strong record of scholarly excellence. Special consideration will be given to candidates with expertise in data mining, security, data management and other analytical methods related to business operations. Teaching and curriculum development at the undergraduate, MBA, and Ph.D. levels will be expected.

Rutgers University is an affirmative action equal opportunity employer. Applications received by February 1, 2011 are guaranteed full consideration. All applicants should have completed a Ph.D. degree in a relevant subject area by the Fall-2011 Semester. Applicants should send curriculum vitae, cover letter, and the names of three references to:

Ms. Carol Gibson
(CGibson@rci.rutgers.edu, pdf files only)
Department of Management Science and Information Systems
Rutgers Business School
Rutgers, The State University of New Jersey
94 Rockafeller Road
Piscataway, NJ 08854-8054

State University of New York at Binghamton
Department of Computer Science
Assistant Professor

We have an opening for a tenure-track Assistant Professor starting Fall 2011. Our preferred specializations are embedded systems, energy-aware computing and systems development. We have well-established BS, MS and PhD programs, with well over 50 full-time PhD students. We offer a significantly reduced teaching load for junior faculty for at least the first three years. Please submit a resume and the names of three references at: http://binghamton.interviewexchange.com

First consideration will be given to applications that are received by January 15, 2011. We are an EE/AA employer.

Stevens Institute of Technology
Three Tenure Track Faculty Positions:
Social Networks, Decision and Cognitive Sciences, and Social Computing

The Howe School of Technology Management at Stevens Institute of Technology announces three tenure-track faculty positions, at either the assistant or associate professor level, for the 2011-2012 academic year.

Social Networks. Candidates should have a background in Information Systems or other fields pertinent to the position. Expertise in social network analysis is expected, and candidates may also have an interest in its applications to financial market behavior.

Decision and Cognitive Sciences. Candidates for should have a background in decision analysis and be knowledgeable about cognitive science and its methods, from work in the disciplines of marketing, management, cognitive science, or psychology. An interest in the applications of decision making is also important, in, for example, the areas of consumer behavior, negotiation and risk and uncertainty.

Social Computing. Candidates should have demonstrated expertise in designing, building and analyzing systems that combine aspects of human and machine intelligence to solve large-scale problems, e.g., organizational, systems, financial. Candidates should also have a background in Information Systems or a related discipline and be cognizant of recent research in crowdsourcing and the cloud computing architectures that underlie it.

Candidates for all positions are expected to have demonstrated the capacity to perform high-impact research. Classroom experience is also expected. Applicants must apply online at http://www.apply2jobs.com/Stevens where they will be asked to create an applicant profile and to formally apply for the position. Use job requisition number MGMT2077 for Social Networks, job requisition number MGMT2078 for Decision and Cognitive Sciences and job requisition number MGMT2079 for Social Computing.

In addition, please send a curriculum vitae, statement of interest in the position, statement of research and teaching interests, three references and a sample of published or other research.

Swarthmore College
Visiting Assistant Professor

Swarthmore College invites applications for a three-year faculty position in Computer Science, at the rank of Visiting Assistant Professor, beginning September 2011. Specialization is open. Review of applications will begin January 1, 2011, and continue until the position is filled. For information, see http://www.cs.swarthmore.edu/job.

Swarthmore College has a strong commitment to excellence through diversity in education and employment and welcomes applications from candidates with exceptional qualifications, particularly those with demonstrable commitments to a more inclusive society and world.

Texas A&M University
Department of Visualization
Assistant Professor

Tenure-track faculty in the area of interactive media. Responsibilities include research/creative work, advising graduate/undergraduate levels, service to dept, university & field, teaching inc. intro courses in game design & development.

Candidates must demonstrate collaborative efforts across disciplinary lines. Graduate degree related to game design & development, mobile media, interactive graphics, interactive art, multimedia or simulation is required. Apply URL: http://www.viz.tamu.edu

The Johns Hopkins University
Tenure-track Faculty Positions

The Department of Computer Science at The Johns Hopkins University is seeking applications for tenure-track faculty positions. The search is open to all areas of Computer Science, with a particular emphasis on candidates with research interests in machine learning, theoretical computer science, computational biology, computational aspects of biomedical informatics, or other data-intensive or health-related applications.

All applicants must have a Ph.D. in Computer Science or a related field and are expected to show evidence of an ability to establish a strong, independent, multidisciplinary, internationally recognized research program. Commitment to quality teaching at the undergraduate and graduate levels will be required of all candidates. Preference will be given to applications at the assistant professor level, but other levels of appointment will be considered based on area and qualifications. The Department is committed to building a diverse educational environment; women and minorities are especially encouraged to apply.

A more extensive description of our search can be found at http://www.cs.jhu.edu/Search2011. More information on the department is available at http://www.cs.jhu.edu.

Applicants should apply using the online application which can be accessed from http://www.cs.jhu.edu/apply. Applications should be received by Dec 1, 2010 for full consideration. Questions should be directed to fsearch@cs.jhu.edu. The Johns Hopkins University is an EEO/AA employer.

Faculty Search
Johns Hopkins University
Department of Computer Science
Room 224 New Engineering Building
Baltimore, MD 21218-2682
Phone: 410-516-8775
Fax: 410-516-6134
fsearch@cs.jhu.edu
http://www.cs.jhu.edu/apply

The Ohio State University
Department of Computer Science and Engineering (CSE)
Assistant Professor

The Department of Computer Science and Engineering (CSE), at The Ohio State University, anticipates significant growth in the next few years. This year, CSE invites applications for four tenure-track positions at the Assistant Professor level. Priority consideration will be given to candidates in database systems, graphics & animation, machine learning, and networking. Outstanding applicants in all CSE areas (including software engineering & programming languages, systems, and theory) will also be considered.

The department is committed to enhancing faculty diversity; women, minorities, and individuals with disabilities are especially encouraged to apply.

Applicants should hold or be completing a Ph.D. in CSE or a closely related field, have a commitment to and demonstrated record of excellence in research, and a commitment to excellence in teaching.

To apply, please submit your application via the online database. The link can be found at: http://www.cse.ohio-state.edu/department/positions.shtml

Review of applications will begin in November and will continue until the positions are filled.

The Ohio State University is an Equal Opportunity/Affirmative Action Employer.

The Pennsylvania State University
Tenure-track faculty

The Department of Computer Science and Engineering (CSE) invites applications for tenure-track faculty positions at all ranks. We seek outstanding candidates who can contribute to the core of computer science and engineering through a strong program of interdisciplinary research in areas such as high performance computing applications and computational modeling for energy, life sciences, environmental sustainability, etc.

The department has 32 tenure-track faculty representing major areas of computer science and engineering. Eleven members of our faculty are recipients of the NSF Career Award. Two faculty members have received the prestigious NSF PECASE Award. In recent years, our faculty received seven NSF ITR Grants, a $35M Network Science Center Award, over $4.5M in computing and research infrastructure and instrumentation grants from NSF, eleven NSF Cyber Trust and Networking awards, and several awards from DARPA, DOE and DoD. There are state-of-the-art research labs for computer systems, computer vision and robotics, Microsystems design and VLSI, networking and security, high performance computing, bioinformatics and virtual environments. The department offers a graduate program with over 40 Masters students and 153 Ph.D. students, and undergraduate programs in computer science and computer engineering. The university is committed to growing the faculty ranks over the next several years and promoting interdisciplinary research toward cyber-enabled discovery and design.

Penn State is a major research university and is ranked 3rd in the nation in industry-sponsored research. Computer science is ranked 6th in the nation in research expenditures. *U.S. News and World Report* consistently ranks Penn State's College of Engineering undergraduate and graduate programs in the top 15 of the nation. As reported in the Chronicles of Higher Education, computer science is ranked 3rd and computer engineering is ranked 8th in the nation, respectively.

The university is located in the beautiful college town of State College in the center of Pennsylvania. State College has 40,000 inhabitants and offers a variety of cultural and outdoor recreational activities nearby. The university offers outstanding events from collegiate sporting events to fine arts productions. Many major population centers on the east coast (New York, Philadelphia, Pittsburgh, Washington D.C., Baltimore) are only a few hours drive away and convenient air services to several major hubs are operated by four major airlines out of State College.

Applicants should hold a Ph.D. in Computer Science, Computer Engineering, or a closely related field and should be committed to excellence in both research and teaching. Support will be provided to the successful applicants for establishing their research programs. We encourage dual career couples to apply. Applications should be received by January 31, 2011 to receive full consideration. To apply by electronic mail, send your resume (including curriculum vitae and the names and addresses of at least three references) as a pdf file to recruiting@cse.psu.edu.

For more information about the Department of CSE at PSU, see http://www.cse.psu.edu. Click here to fill out an Affirmative Action Applicant Data Card. Our search number is **015-87**. You **MUST** include this search number in order to submit this form.

Penn State is committed to affirmative action, equal opportunity and the diversity of its workforce.

The University of Alabama at Birmingham
Assistant/Associate Professor

The Department of Computer & Information Sciences at the University of Alabama at Birmingham (UAB) is seeking candidates for a tenure-track/tenure-earning faculty position at the Assistant or Associate Professor level beginning August 15, 2011.

Candidates with leading expertise in Information Assurance, particularly Computer Forensics and/or Computer and Network Security are sought. The successful candidate must be able to participate effectively in multidisciplinary research with scientists in Computer and Information Sciences and Justice Sciences for advancing Information Assurance Research at UAB, including joint scientific studies, co-advising of students, and funding. Allied expertise in Artificial Intelligence, Knowledge Discovery and Date Mining, Software Engineering, and/or High Performance Computing is highly desirable. UAB has made significant commitment to this area of research and teaching. Candidates must consequently have strong teaching credentials as well as research credentials.

For additional information about the department please visit http://www.cis.uab.edu.

Applicants should have demonstrated the potential to excel in one of these areas and in teaching at all levels of instruction. They should also be committed to professional service including departmental service. A Ph.D. in Computer Science or closely related field is required.

Applications should include a complete curriculum vita with a publication list, a statement of future research plans, a statement on teaching experience and philosophy, and minimally two letters of reference with at least one letter addressing teaching experience and ability.

Applications and all other materials may be submitted via email to facapp.ia@cis.uab.edu or via regular mail to:

Search Committee
Department of Computer and Information Sciences
115A Campbell Hall
1300 University Blvd
Birmingham, AL 35294-1170

Interviewing for the position will begin as soon as qualified candidates are identified, and will continue until the position is filled.

The department and university are committed to building a culturally diverse workforce and strongly encourage applications from women and individuals from underrepresented groups. UAB has a Spouse Relocation Program to assist in the needs of dual career couples. UAB is an Affirmative Action/Equal Employment Opportunity employer.

Theophilus, Inc.
Recommendation Engine / Java Developer

Funded startup with virtual office and flexible working hours seeking experienced part-time Java Developer with recommendation engine experience (implementation and experimental evaluation), to work on next generation recommendation product integrating advanced semantic analysis, search, social networks, and smartphones (iPhone). Developer can work remotely.

Email for full description: david.kim@theophilus-inc.com

Toyota Technological Institute at Chicago
Computer Science Faculty Positions at All Levels

Toyota Technological Institute at Chicago (TTIC) is a philanthropically endowed degree-granting institute for computer science located on the University of Chicago campus. The Institute is expected to reach a steady-state of 12 traditional faculty (tenure and tenure track), and 12 limited term faculty. Applications are being accepted in all areas, but we are particularly interested in:

Theoretical computer science
Speech processing
Machine learning
Computational linguistics
Computer vision
Computational biology
Scientific computing

Positions are available at all ranks, and we have a large number of limited term positions currently available.

For all positions we require a Ph.D. Degree or Ph.D. candidacy, with the degree conferred prior to date of hire. Submit your application electronically at:

http://ttic.uchicago.edu/facapp/

Toyota Technological Institute at Chicago is an Equal Opportunity Employer

University of California, Irvine
Computer Science Department
Tenure-Track Position in Operating Systems / Programming Languages

Department of Computer Science at the University of California, Irvine (UCI) invites applications for a tenure-track Assistant Professor position in the general area of Systems. We are particularly interested in applicants who specialize in Operating Systems, Programming Languages or Distributed Systems. Exceptionally qualified more senior candidates may also be considered.

Department of Computer Science is the largest department in the Bren School of Information and Computer Sciences, one of only a few such schools in the nation and the only one in the UC System. The department has over 45 faculty members and over 200 PhD students. Faculty research is very vibrant and broad, spanning prominent areas such as: distributed systems, software, networking, databases, embedded systems, theory, security, graphics, multimedia, machine learning, AI, and bioinformatics. Prospective applicants are encouraged to visit our web page at: http://cs.www.uci.edu/

One of the youngest UC campuses, UCI is ranked 10th among the nation's best public universities by US News & World Report. It has received three Nobel prizes in the past 15 years. Salary and other compensation (including priority access to on-campus for-sale faculty housing) are competitive with the nation's finest universities. UCI is located 3 miles from the Pacific Ocean in Southern California (50 miles South of Los Angeles) with a very pleasant year-round climate. The area offers numerous recreational and cultural opportunities. Also, the Irvine public school system is one of the highest-ranked in the nation.

Screening will begin immediately upon receipt of a completed application. Applications will be accepted until the position is filled, although maximum consideration will be given to applications received by December 15, 2010. Each application must contain: a cover letter, CV, sample publications (up to 3) and 3-5 letters of recommendation. All these materials must be uploaded on-line. Please refer to the following web site for instructions:

https://recruit.ap.uci.edu/

UCI is an equal opportunity employer committed to excellence through diversity and encourages applications from women, minorities, and other under-represented groups. UCI is responsive to the needs of dual career couples, is dedicated to work-life balance through an array of family-friendly policies, and is the recipient of an NSF Advance Award for gender equity.

University of California, Riverside
Tenure-Track Faculty Positions

The Department of Computer Science and Engineering, University of California, Riverside invites applications for **tenure-track** faculty positions beginning in the 2011/2012 academic year with research interests in **(a) Operating and Distributed Systems (b) Data Mining and, (c) Computer Graphics. Exceptional candidates in all areas will be considered.** A Ph.D. in Computer Science (or in a closely related field) is required at the time of employment. Junior candidates must show outstanding research, teaching and graduate student mentorship potential. Exceptional senior candidates may be considered. Salary level will be competitive and commensurate with qualifications and experience. Details and application materials can be found at www.engr.ucr.edu/facultysearch. Full consideration will be given to applications received by February 1, 2011. Applications will continue to be received until the positions are filled. For inquiries and questions, please contact us at search@cs.ucr.edu. EEO/AA employer.

University of Houston – Clear Lake
Assistant or Associate Professor of Computer Science/Computer Information Systems

The Computer Science and Computer Information Systems programs of the School of Science and Computer Engineering at the University of Houston-Clear Lake invite applications for tenure-track Assistant or Associate Professor of CS or CIS to begin August 2011. Ph.D. in CS, CIS/IS, or a closely related field is required. Applications are accepted only online at https://jobs.uhcl.edu. See http://sce.uhcl.edu/cs and http://sce.uhcl.edu/cis for additional information about CS/CIS programs. AA/EOE.

University of Houston-Downtown
Assistant Professor, Computer Sciences

The Department of Computer and Mathematical Sciences invites applications for a tenure-track Assistant Professor position in Computer Science starting Fall 2011. Successful candidates will have a PhD in Computer Science or a closely related field in hand by the time of appointment, a promising research profile, and a commitment to excellence in teaching. Review of applications will begin immediately and continue until the position is filled. Only online applications submitted through http://jobs.uhd.edu will be considered.

University of Mississippi
Chair, Department of Computer and Information Science

The Department of Computer and Information Science at the University of Mississippi (Ole Miss) invites applications for the position of Chair. The Chair provides leadership and overall strategic direction for the instructional and research programs. Requirements include a PhD or equivalent in computer science or a closely related field, evidence of excellence in teaching and research in one or more major areas of computer and information science, and administrative experience relevant to the management of an academic computer science department. The Department has an ABET/CAC-accredited undergraduate program and MS and PhD programs. See the website http://www.cs.olemiss.edu for more information about the Department and its programs.

The University is located in the historic town of Oxford in the wooded hills of north Mississippi, an hour drive from Memphis. Oxford has a wonderful small-town atmosphere with affordable housing and excellent schools.

Requirements include a PhD or equivalent in computer science or a closely related field, evidence of excellence in teaching and research in one or more major areas of computer and information science, and administrative experience relevant to the management of an academic computer science department.

Individuals may apply online at http://jobs.olemiss.edu. Applicants will be asked to upload a cover letter, curriculum vitae, names and contact information for five references, and a statement of department administrative philosophy, objectives, and vision. Review of applications will begin immediately and will continue until the position is filled or an adequate applicant pool is reached.

The University of Mississippi is an EEO/AA/Title VI/Title IX/Section 504/ADA/ADEA employer.

University of North Texas
Department of Computer Science and Engineering
Department Chair

Applications are invited for the Chair position in the Department of Computer Science and Engineering at the University of North Texas. UNT is one of seven universities designated by the state as an "Emerging Research University." Candidates must have an earned doctorate in Computer Science and Engineering or a closely related field with a record of significant and sustained research funding and scholarly output that qualifies them to the rank of full professor. Preferred: Administrative experience as a department chair or director of personnel working in computer science and engineering; experience in curriculum development; and demonstrated experience mentoring junior faculty. The committee will begin its review of the applications on December 1, 2010 and the position will close on April 4, 2011. All applicants must apply online to: https://facultyjobs.unt.edu. Nominations and any questions regarding the position may be directed to Dr Bill Buckles (bbuckles@cse.unt.edu). Additional information and about the department is available at www.cse.unt.edu. UNT is an AA/ADA/EOE.

University of Texas-Pan American
Computer Science Department
Assistant Professor Faculty Position

The Department of Computer Science at the University of Texas-Pan American (UTPA) seeks applications for a tenure-track Assistant Professor position in Computer Engineering (F10/11-24). All candidates must have a potential/proven record in teaching and active research. The Assistant Professor position in Computer Engineering requires a Ph.D. in computer science or computer engineering. Highest priority will be given to candidates with research expertise in areas of Computer Networks, Cybersecurity/Forensics, Data Management, Scientific Workflows, and/or Semantic Web.The program in Computer Engineering leading to BS degree in Computer Engineering is administered jointly by the Computer Science Department and the Electrical Engineering Department. The Computer Science Department (http://www.cs.panam.edu) also offers the BSCS (ABET/CAC Accredited) and BS undergraduate degrees, MS in Computer Science and MS in Information Technology.

UTPA is situated in the lower Rio Grande valley of south Texas, a strategic location at the center of social and economic change. With a population of over one million, the Rio Grande Valley is one of the fastest growing regions in the country. The region has a very affordable cost-of-living. UTPA is a leading educator of Hispanic/Latino students, with enrollment of over 18,500.

The position starts in Fall 2011. The salary is competitive. A complete application should include: (1) a cover letter, specifically stating an interest in the Assistant Professor in Computer Engineering position, noting your specialization, (2) vita, (3) statements of teaching and research interests, and (4) names and contact information of at least three references. Applications can be mailed to Dean's Office, Computer Engineering Search, College of Engineering and Computer Science, The University of Texas-Pan American, 1201 W. University Drive, Edinburg, Texas 78539 or emailed to coec@utpa.edu. Review of materials will begin on November 1, 2010 and continue until the position is filled.

NOTE: UTPA is an Equal Opportunity/Affirmative Action employer. Women, racial/ethnic minorities and persons with disabilities are encouraged to apply. This position is security-sensitive as defined by the Texas Education Code §51.215(c) and Texas Government Code §411.094(a)(2). Texas law requires faculty members whose primary language is not English to demonstrate proficiency in English as determined by a satisfactory grade on the International Test of English as a Foreign Language (TOEFL).

University of Wisconsin-Platteville
Assistant Professor

The University of Wisconsin-Platteville Computer Science and Software Engineering Department has two tenure track positions to be filled in Fall 2011. One is in Software Engineering and one is an anticipated opening in Computer Science. For more information and to apply electronically: http://www.uwplatt.edu/csse/positions.

Utah State University
Assistant Professor

Applications are invited for a faculty position at the Assistant Professor level, for employment beginning Fall 2011. Applicants must have completed a PhD in computer science by the time of appointment. The position requires demonstrated research success, a significant potential for attracting external research funding, excellence in teaching both undergraduate and graduate courses, the ability to supervise student research, and excellent communication skills.

USU offers competitive salaries and outstanding medical, retirement, and professional benefits (see http://www.usu.edu/hr/ for details). The department currently has approximately 280 undergraduate majors, 80 MS students and 27 PhD students. There are 17 full time faculty. The BS degree is ABET accredited. Utah State University is a Carnegie Research Doctoral extensive University of over 23,000 students, nestled in a mountain valley 80 miles north of Salt Lake City, Utah. Opportunities for a wide range of outdoor activities are plentiful. Housing costs are at or below national averages, and the area provides a supportive environment for families and a balanced personal and professional life. Women, minority, veteran and candidates with disabilities are encouraged to apply. USU is sensitive to the needs of dual-career couples. Utah State University is an affirmative action/equal opportunity employer, with a National Science Foundation ADVANCE Gender Equity program, committed to increasing diversity among students, faculty, and all participants in university life.

Applications must be submitted using USU's online job-opportunity system. To access this job opportunity directly and begin the application process, visit https://jobs.usu.edu/applicants/Central?quickFind=54615.

The review of the applications will begin on January 15, 2011 and continue until the position is filled. The salary will be competitive and depend on qualifications.

Wichita State University
Assistant Professor

The Department of Electrical Engineering and Computer Science at Wichita State University has multiple open tenure-eligible faculty positions at the assistant professor level in Electric Energy Systems, Information Security, and Software Engineering. Duties and responsibilities of all positions include teaching undergraduate and graduate courses, advising undergraduate students, supervising MS and PhD students in their theses and dissertations, obtaining research funding, conducting an active research program, publishing the results of research, actively participating in professional societies, and service to the department, college and university. Complete information can be found on our website, www.eecs.wichita.edu.

To ensure full consideration, the complete application package must be submitted online at jobs.wichita.edu by January 15, 2011. Applications will be continuously reviewed after that date until determinations are made with regard to filling the positions. Offers of employment are contingent upon completion of a satisfactory criminal background check as required by Board of Regents policy.

Questions only (not applications) can be directed to the search chair, Ward Jewell, wardj@ieee.org.

Wichita State University is an equal opportunity and affirmative action employer.

[CONTINUED FROM P. 128] as calculators for business and scientific applications. We were using it, however, as a personal augmenter. The hypertext system we built was used in the early 1970s for a poetry class, where someone would enter a poem and invite commentary about it. This was like creating a wiki, only 35 years earlier.

How different is it teaching computer science now than it was five, 10, or 20 years ago, given how fast-shifting the landscape is?

Actually, much has stayed the same. We're still teaching students the process of discovery. Children have an innate interest in discovering everything around them. But, tragically, by the time they've gotten to college, this has usually been beaten out of them. So we teach them how to be children again—how to think like they did when they were four years old.

Undergraduates go to a research university to engage in discovery. We constantly convey to them that we all got involved in computer science because we wanted to solve real problems and change the world. To do this in the modern era just makes it more exciting. The students in my classroom now will go on to computer science grad school, to law school, to medical school, to Microsoft, Google, Amazon, and startups. Then, they change the world.

What was your philosophy when it came to chairing the department?

First, it's about leadership, not administration. Be a leader; hire an administrator. Second, it's about people and making them productive. You provide a shield that allows the people working for you to focus on their research and teaching. Third, it's about students—they're the multiplicative factor. If you're at a university and you're not engaged with students, you're in the wrong line of work—go to an industrial research lab.

With respect to your role in the eScience Institute, how are you working to take research about disruptive technologies and make a real-world impact?

We are undergoing a revolution in science. The growth of data sensors has been phenomenal. They're everywhere—in cell phones, in telescopes, in gene sequencers, in highways and buildings and bridges, on the sea floor, in forest canopies, on the Web. In the old days, if you wanted to study the creation of a social clique, for example, you'd pay some undergraduates $6 to participate in a focus group during their lunch break. Now, you have millions of such cliques that can be researched through social media. The challenge of research before was finding enough data. Now, we're drowning in it. This is a new turn of the crank in the field of computational science—it's about the data, not the cycles.

"I'm surrounded by people who can be counted on to produce. I try to be one of those people, too."

With the eScience Institute, we're focusing on finding better ways to pursue the collection and exploration of all of this data. We want to apply this knowledge first to the needs of the research scientists here at the university—the biologists and astrophysicists who need to resolve big-data exploration problems to do their jobs. Then, we can create solutions for people around the world.

You're also chair of the Computing Community Consortium (CCC). What are the primary efforts the CCC is most involved with now? And what do you hope to accomplish with these efforts?

Let's face it, as computer scientists, we can come across as a bit geeky. Our neighbors don't understand what we do. But what we do greatly impacts the issues that affect our neighbors, like the improvement of our transportation systems, energy efficiency, health care, education, open government, cybersecurity, and discovery in all fields. As computer scientists, we have a rich intellectual agenda with the capability to have an enormous breadth of impact upon daily lives. We need to do a better job of telling that story, and a better job of garnering the resources to do the research that will enable further breakthroughs. That's what the CCC seeks to do.

You also seek to increase the role of computer science in K–12, within the Science, Technology, Engineering, and Mathematics (STEM) Education Coalition learning framework. Kids spend plenty of time with computers these days, but it seems more voyeuristically focused, with lots of social networking and watching YouTube clips. How do you get them to take a step beyond?

There's a positive aspect to what they're doing. It's better than in the days when the only computer in the house was a video game console. When you have young people using their home computers or iPhones for communication and social networking, it's good because it introduces them to the broader power of computer science. Hopefully, many of these young people will say, "I want to create something like this." But to empower these kids, there needs to be a revolution in STEM education.

All available data tells us that our K–12 students are falling behind the rest of the developed world. To address this, we're exploring how to better teach STEM disciplines. We need to revive the concept that science is about discovery, and not the memorization of facts. And computer science needs to be a big part of this revolution—it needs to be viewed as an essential STEM discipline.

You clearly have a lot on your plate. How do you serve so many needs without shortchanging some?

I'm the wrong person to ask! As responsibilities increase, the hours of the day don't. If I'm writing a report, I always want to have another month. If I'm preparing for a class, I always want to have another hour. I wish I had to power to say "no," but I don't. I'm surrounded by people who can be counted on to produce. I try to be one of those people, too.

Dennis McCafferty is a Washington, D.C.-based technology writer.

DOI:10.1145/1866739.1866763

Dennis McCafferty

Q&A
A Journey of Discovery

Ed Lazowska discusses his heady undergraduate days at Brown University, teaching, eScience, and being chair of the Computing Community Consortium.

AS AN UNDERGRADUATE student at Brown University, Ed Lazowska hardly seemed destined to become a leader in computer science. Actually, he wasn't sure what he wanted to do. He started as an engineering student, switched to physics, and briefly considered chemistry. Essentially, he was "adrift." (His description, not ours.)

It wasn't until he fell under the tutelage of computer science professor Andy van Dam that he discovered what really excited him: the process of discovery.

"We had access to an IBM 360 mainframe that occupied an entire building," Lazowska recalls. "Despite its size, it had only a couple hundred megabytes of disk storage and 512 kilobytes of memory. Today, your typical smartphone will have 1,000 times the processing power and storage of this machine. During the day, it supported the entire campus. But between midnight and 8 A.M., we were allowed to use it as a personal computer. We were building a 'what-you-see-is-what-you-get' hypertext editor—Microsoft Word plus the Web, minus networking. It was revolutionary."

Four decades later, Lazowska is dedicated to making the same transformational impact on countless computer science students. After graduating from Brown in 1972, he received his Ph.D. from the University of Toronto in 1977, and joined the University of Washington faculty, focusing on the design, implementation, and analysis of high-performance computing and communication systems. Today, Lazowska holds the Bill & Melinda Gates Chair in Computer Science & Engineering at the University of Washington, where he served as department chair from 1993 to 2001. He also directs the university's eScience Institute and chairs the Computing Community Consortium, a National Science Foundation initiative that seeks to inspire computer scientists to tackle the societal challenges of the 21st century.

How invigorating were those early days at Brown under van Dam?

It was an amazing time. He had a crew of 20 undergraduates who were his research assistants. Typically, research assistants were graduate students, but Brown had few computer science graduate students at the time. Andy was asking us to join him in discovery—to figure out how to do things that no one had done before. Up to that time, including my freshman year at Brown, I had been learning things that people already knew. It blew my mind that Andy was asking 19- and 20-year-olds to find answers to questions that he himself didn't know. People rise to the expectations and challenges that are set for them. Andy understood this.

Back then, most people thought of computers [CONTINUED ON P. 127]

PHOTOGRAPH BY BRIAN SMALE

IEEE 7th World Congress on Services (SERVICES 2011)

July 5-10, 2011, Washington DC, USA, http://www.servicescongress.org/2011

Modernization of all vertical services industries including finance, government, media, communication, healthcare, insurance, energy and ...

IEEE 8th International Conference on Services Computing (SCC 2011)

In the modern services and software industry, Services Computing has become a cross-discipline that covers the science and technology of bridging the gap between Business Services and IT Services. The scope of Services Computing covers the whole lifecycle of services innovation research that includes business componentization, services modeling, services creation, services realization, services annotation, services deployment, services discovery, services composition, services delivery, service-to-service collaboration, services monitoring, services optimization, as well as services management. The goal of Services Computing is to enable IT services and computing technology to perform business services more efficiently and effectively. Visit http://conferences.computer.org/scc.

IEEE 9th International Conference on Web Services (ICWS 2011)

As a major implementation technology for modernizing software and services industry, Web services are Internet-based application components published using standard interface description languages and universally available via uniform communication protocols. The program of ICWS 2011 will continue to feature research papers with a wide range of topics focusing on various aspects of implementation and infrastructure of Web-based services. ICWS has been a prime international forum for both researchers and industry practitioners to exchange the latest fundamental advances in the state of the art on Web services. Visit icws.org.

IEEE 4th International Conference on Cloud Computing (CLOUD 2011)

Cloud Computing is becoming a scalable services delivery and consumption platform in the field of Services Computing. The technical foundations of Cloud Computing include Service-Oriented Architecture (SOA) and Virtualizations of hardware and software. The goal of Cloud Computing is to share resources among the cloud service consumers, cloud partners, and cloud vendors in the cloud value chain. Major topics cover Infrastructure Cloud, Software Cloud, Application Cloud, and Business Cloud. Visit http://thecloudcomputing.org.

Sponsored by IEEE Technical Committee on Services Computing (TC-SVC, tab.computer.org/tcsc)

IEEE

IEEE computer society

Services Computing

IBM

SAP

IBM Research

Submission Deadlines

ICWS 2011: 1/31/2011
CLOUD 2011: 1/31/2011
SCC 2011: 2/14/2011
SERVICES 2011: 2/14/2011

Contact: Liang-Jie Zhang (LJ) at zhanglj@ieee.org
(Steering Committee Chair)

IEEE TRANSACTIONS ON SERVICES COMPUTING

ACM SIGCHI TEI 2011

Call for Participation

The International Conference on Tangible, Embedded and Embodied Interaction addresses HCI issues, design, interactive art, user experience, tools and technologies, with a focus on how computing can finally bridge atoms and bits into cohesive interactive systems.

TEI is a rapidly growing, single-track, demo-friendly, peer-reviewed conference. It provides a forum for exchanging ideas and presenting innovative work through talks, interactive exhibits, demos, hands-on studios, posters, art installations and performances.

http://www.tei-conf.org

Register online before January 19 or on site.

TEI '11

Fifth International Conference on Tangible, Embedded and Embodied Interaction

January 23 - Studios and Workshops
January 24 to 26 - Technical Program

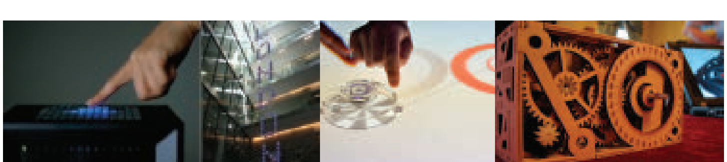
Madeira Interactive Technologies Institute
Madeira, Portugal.

General Chairs
Mark D Gross, Nuno Jardim Nunes

Program Chairs
Ellen Yi-Luen Do, Stephen Brewster, Ian Oakley

I love TEI because it's where great designers and engineers present their work in its truest, most honest form.

Leah Buechley
High-Low Tech Group, MIT Media Lab



The more I learn about the TEI conference, the more I like it. In fact, that is an understatement: the more I learn about it, the more I believe it essential.

Don Norman
Nielsen Norman Group