Experience the Newest and Most Advanced Thinking in Big Data Analytics

03 November 2016 | Austin, TX

Big Data: Big Hype or Big Imperative?

BOTH.

Business departments know the promise of big data—and they want it! But newly minted data scientists can’t yet meet expectations, and technologies remain immature. Yes, big data is transforming the way we do—everything. But knowing that doesn’t help you decide what steps to take tomorrow to assure your company’s future.

That’s why May 24 is your real-world answer. Come meet the experts who are grappling with and solving the problems you face in mining the value of big data. You literally can’t afford to miss the all new Rock Stars of Big Data 2016.

www.computer.org/bda

Rock Star Speakers

Kirk Borne
Principal Data Scientist, Booz Allen Hamilton

Satyam Priyadarshy
Chief Data Scientist, Halliburton

Bill Franks
Chief Analytics Officer, Teradata
Cdoe Obofsucaitn: Securing Software from Within

Gambling with Source Code

Developing Dependable and Secure Cloud Applications
<table>
<thead>
<tr>
<th>Page</th>
<th>Title</th>
<th>Authors/Contributors</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Spotlight on Transactions: Digital-Microfluidics Biochips</td>
<td>Mohamed Ibrahim and Krishnendu Chakrabarty</td>
</tr>
<tr>
<td>8</td>
<td>Editor’s Note: Note: Keeping up with the Ever-Changing Software Industry</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Cdoe Obosucaitn: Securing Software from Within</td>
<td>Sebastian Lins, Pascal Grochol, Stephan Schneider, and Ali Sunyaev</td>
</tr>
<tr>
<td>20</td>
<td>A Guide to Cloud-Enabling Your Software</td>
<td>David S. Linticum</td>
</tr>
<tr>
<td>24</td>
<td>Gambling with Source Code</td>
<td>Nir Kshetri</td>
</tr>
<tr>
<td>28</td>
<td>Being a DevOps Developer</td>
<td>Diomidis Spinellis</td>
</tr>
<tr>
<td>30</td>
<td>Developing Dependable and Secure Cloud Applications</td>
<td>Ingo Weber, Surya Nepal, and Liming Zhu</td>
</tr>
<tr>
<td>36</td>
<td>Frequently Unanswered Questions</td>
<td>Gerard J. Holzmann</td>
</tr>
<tr>
<td>40</td>
<td>The Role of Data Science in Web Science</td>
<td>Christopher Pethenan, Elena Simperl, Thanassis Tiropanis, Ramine Tinati, and Wendy Hall</td>
</tr>
<tr>
<td>46</td>
<td>Cyber-Physical-Human Systems: Putting People in the Loop</td>
<td>Sulayman K. Sowe, Eric Simmon, Koji Zettsu, Frederic de Vaulx, and Irena Bojanova</td>
</tr>
<tr>
<td>50</td>
<td>What Does a Skunk Works Do?</td>
<td>Shane Greensstein</td>
</tr>
</tbody>
</table>

Departments

- 6 Magazine Roundup
- 60 Career Opportunities

Subscribe to *ComputingEdge* for free at [www.computer.org/computingedge](http://www.computer.org/computingedge).
Digital-Microfluidic Biochips

Mohamed Ibrahim and Krishnendu Chakrabarty, Duke University

This installment highlighting the work published in IEEE Computer Society journals comes from IEEE Transactions on Multi-Scale Computing Systems.

Digital-microfluidic biochips (DMFBs) are revolutionizing laboratory procedures for point-of-care clinical diagnostics, environmental monitoring, and drug discovery. DMFBs allow bioassay protocols to be scaled down to droplet size. They’re executed by enabling precise control of discrete droplets using a patterned array of electrodes.

Biochemistry’s inherent complexity means that operational errors due to unbalanced splitting or protein fouling might arise during bioassay execution (see Figure 1a). Therefore, a core challenge in operating DMFBs is to verify the correctness of fluidic interactions; thus, an efficient DMFB design will require careful consideration of error recovery. In “Efficient Error Recovery in Cyberphysical Digital-Microfluidic Biochips” (IEEE Trans. Multi-Scale Computing Systems, vol. 1, no. 1, 2015, pp. 46–58), we propose an online synthesis framework that supports error recovery in pin-constrained DMFBs and provide a comprehensive analysis of error recoverability.

Given a general-purpose pin-constrained DMFB configuration connected to a real-time charge-coupled device camera system, we employed a dynamic adaptation technique to generate new schedules, placements, and droplet routes in response to errors (see Figure 1b).

The use of our framework avoids faulty components and, thus, ensures the reliability of DMFB operation. Error-recoverability analysis determines the amount of chip
enabling precise control of discrete to droplet size. They’re executed by bioassay protocols to be scaled down ining, and drug discovery. DMFBs allow diagnostics, environmental monitor-

resources required for error recovery given completion-time constraints. Such analysis can help determine how large a biochip should be for a target application.

MOHAMED IBRAHIM is a PhD student in the Department of Electrical and Computer Engineering at Duke University. Contact him at mohamed.s.ibrahim@duke.edu.

KRISHNENDU CHAKRABARTY is the William H. Younger Distinguished Professor of Engineering in the Department of Electrical and Computer Engineering and a professor of computer science at Duke University. Contact him at krish@ee.duke.edu.

Figure 1. Online synthesis framework to support error recovery in pin-constrained digital-microfluidic biochips. (a) Operational-error mechanisms. (b) Proposed workflow: the control software keeps track of the status of each bioassay operation and reacts to errors by feeding the online synthesis algorithm with required “tokens” for adaptation. CCD: charge-coupled device.

For more information, including a list of compatible devices, visit www.computer.org/digital-library, and you can read them on any device that supports ePub. Just download the articles from the IEEE Computer Society Digital Library, and you can read them on any device that supports ePub.

IEEE Computer Society journals in IEEE Computer Society magazines and Transactions are now available to subscribers in the portable ePub format.

This installment highlighting the work published in IEEE Computer Society journals comes from Computing Systems.


MOHAMED IBRAHIM and Krishnendu Chakrabarty, Duke University

Digital-Microfluidic Biochips

Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.

2017 B. Ramakrishna Rau Award Call for Nominations

Honoring contributions to the computer microarchitecture field

New Deadline: 1 May 2017

Established in memory of Dr. B. (Bob) Ramakrishna Rau, the award recognizes his distinguished career in promoting and expanding the use of innovative computer microarchitecture techniques, including his innovation in compiler technology, his leadership in academic and industrial computer architecture, and his extremely high personal and ethical standards.

AWARD: Certificate and a $2,000 honorarium.


NOMINATION SUBMISSION: This award requires 3 endorsements. Nominations are being accepted electronically: www.computer.org/web/awards/rau

CONTACT US: Send any award-related questions to awards@computer.org

www.computer.org/awards
The IEEE Computer Society’s lineup of 13 peer-reviewed technical magazines covers cutting-edge topics ranging from software design and computer graphics to Internet computing and security, from scientific applications and machine intelligence to cloud migration and microchip manufacturing. Here are highlights from recent issues.

**Computer**

As our reliance on information and communications technology increases and attacks become more sophisticated, cybersecurity is more critical and challenging than ever before. This is the focus of *Computer*’s June 2016 special issue on security threats.

**IEEE Software**

The next generation of **software-intensive systems** will be taught instead of programmed. This poses challenges to practitioners developing, delivering, and evolving these systems, according to Grady Booch in “It Is Cold. And Lonely.” from *IEEE Software*’s May/June 2016 issue.

**IEEE Internet Computing**

The authors of “Developing **Dependable and Secure Cloud Applications**,” from *IEEE Internet Computing*’s May/June 2016 issue, analyze the security and dependability challenges that cloud applications face, and present their research into overcoming some of these obstacles.

**Computing in Science & Engineering**

Future high-performance computing centers will expand their roles as service providers. In the process, the facilities will have to focus on meeting their users’ needs as much as on operating machines reliably. “Expanding the Scope of High-Performance Computing Facilities,” from *CiSE*’s May/June 2016 issue, present five interrelated topic areas essential to expanding the value the centers provide to those performing computational science.

**IEEE Security & Privacy**

Users are increasingly adopting traditional and innovative security measures to protect valuable cyberinformation. As a result, new participants are emerging alongside traditional ones. In light of these developments, *IEEE S&P*’s May/June 2016 special issue discusses how interaction approaches, business models, and organizational practices relate to cyber-security economics.
**IEEE Cloud Computing**

The Internet of Things (IoT) has helped advance the goal of getting all devices connected online and providing smart environments worldwide. However, many smart devices’ limited computational capabilities cause concern about their ability to provide the necessary security. *IEEE Cloud Computing’s* March/April 2016 issue provides a platform for academia and industry to share their contributions to a secure and dependable IoT in collaboration with resource-rich cloud infrastructures.

**IEEE Computer Graphics and Applications**

The healthcare industry’s wide-spread digitization is reshaping one of the world’s largest economic sectors. This transformation is helping doctors, researchers, and patients. Given the scale and complexity of the data involved, advanced visualization tools have the potential to play a critical role in this process, as discussed in “Data-Driven Healthcare: Challenges and Opportunities for Interactive Visualization,” from *IEEE CG&A’s* May/June 2016 issue.

**IEEE Intelligent Systems**

*IEEE Intelligent Systems’* May/June 2016 special issue is the second part of a series on pattern recognition. This issue reports on advances in pattern recognition for visual data. The articles address topics such as using cross-domain dictionary learning for visual categorization, matching photos to facial sketches, measuring heartbeat rates from facial videos, and detecting vehicles at night.

**IEEE MultiMedia**

In “Understanding Multimedia,” from *IEEE MultiMedia’s* April–June 2016 issue, magazine editor in chief Yong Rui reflects on how much AI and multimedia technologies have advanced in the past 60 years. He discusses the progress made in understanding visual media and the challenges that lie ahead for video-to-text technology.

**IEEE Annals of the History of Computing**

The articles in *IEEE Annals’* April–June 2016 special issue on the history of computing in East Asia cover real-world computing implementations or applications in the region. The topics include the COMTRAC (computer-aided train traffic control) system that has run the high-speed, high-frequency operations of Japan’s Tokaido Shinkansen railway since 1972; and Korean officials’ controversial effort from 1987 to 1995 to standardize the digital representation of their language’s characters.

**IEEE Pervasive Computing**

Interest is increasing in mobile personal health technologies. To help designers make informed and well-articulated design decisions, the authors of “The Personal Health Technology Design Space,” from *IEEE Pervasive Computing’s* April–June 2016 issue, propose an approach comprising 10 dimensions related to the design of data-sampling strategies, visualization and feedback approaches, treatment models, and regulatory constraints.

**IT Professional**

IT’s importance to healthcare continues to grow, driven by disruptive changes including financial pressures, an aging population, the spread of connectivity and mobile technology, and medical advances. These changes demand improvements in health IT. *IT Pro’s* May/June 2016 special issue on IT trends in healthcare discusses the most recent developments in this area.

**IEEE Micro**

*IEEE Micro’s* May/June 2016 special issue features top papers from the various 2015 computer-architecture conferences, as selected by a committee of 33 leading computer architects from academia and industry.

**Computing Now**

The Computing Now website (http://computingnow.computer.org) features up-to-the-minute computing news and blogs, along with articles ranging from peer-reviewed research to opinion pieces by industry leaders.
Software is a dynamic field, and it’s imperative that those in the industry keep up. This can be a challenge, though, because of the many rapid changes taking place. To help cope with the challenge, this ComputingEdge issue looks at some important software developments and where they might be headed.

In IEEE Software’s “Leaders of Tomorrow on the Future of Software Engineering: A Roundtable,” nine rising stars describe how research in the field will evolve and highlight emerging opportunities and solutions.

“Code Obfuscation: Securing Software from Within,” from IEEE Security & Privacy, discusses how computer scientists and engineers are working to bring code obfuscation from a wobbly art to a sound methodology.

Many vendors are creating cloud-based versions of their products to try to better compete in the complex software marketplace. “A Guide to Cloud-Enabling Your Software,” which appears in IEEE Cloud Computing, discusses the challenges this entails.

Technology companies required to comply with controversial regulations in foreign countries as a cost of doing business there must trade off the potential economic benefits with the risk of long-term problems, according to Computer’s “Gambling with Source Code.”

DevOps is firmly established in many IT sectors, enabling the efficient delivery of higher-quality services. “Being a DevOps Developer,” from IEEE Software, helps software professionals make the most of this approach.

In IEEE Internet Computing’s “Developing Dependable and Secure Cloud Applications,” the authors analyze some of the challenges involved in designing cloud applications and review their research into overcoming these obstacles.

ComputingEdge articles on topics other than software include the following:

- “The Role of Data Science in Web Science,” from IEEE Intelligent Systems, notes that as a discipline to use within Web science research, data science offers opportunities to uncover trends in large Web-based datasets.
- IT Professional’s “Cyber-Physical-Human Systems: Putting People in the Loop” outlines the challenges of integrating people into a new generation of cyber-physical-human systems and proposes a model to help with this process.
- “What Does a Skunk Works Do?” from IEEE Micro examines four types of skunk works—experimental, innovative laboratories or departments within a company or institution—and discusses the issues they face and the benefits they can offer.
Leaders of Tomorrow on the Future of Software Engineering
A Roundtable

Nine rising stars in software engineering shared their perspectives on the future of software engineering at the Leaders of Tomorrow Symposium at the 23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering. Here they describe how software engineering research will evolve, highlighting emerging opportunities and groundbreaking solutions. They predict the rise of end-user programming, the monitoring of developers through neuroimaging and biometrics sensors, analysis of data from unstructured documents, the mining of mobile marketplaces, and changes to how we create and release software. Enjoy! —Massimiliano Di Penta, Ahmed E. Hassan, and Thomas Zimmermann, symposium chairs

In the Future, Everyone Will Be a Programmer for 15 Minutes
Felienne Hermans

IN THE PAST, software engineering has focused mainly on professional developers: people employed to build, test, and maintain software. However, many people program not as a job but as a means to an end. These workers, often called end-user programmers, write queries, small scripts, or spreadsheets to support their daily jobs. The number of end-user programmers in the US alone is estimated at 11 million, compared to only 2.75 million professional programmers\(^1\) Given the popularity of introductory-programming initiatives such as code.org and end users’ widespread adoption of programming languages such as Python and R, we can assume that a new generation of professionals will emerge who can perform some programming to reach their professional goals.

So, supporting end-user programmers in building reliable software will be an even bigger opportunity for research in the near future.

Because end-user programmers create programs to support their own domain of expertise, the programs they create are, by definition, not meant for others to use. However, a core problematic aspect of end-user programming is that sometimes the created artifacts evolve from personal solutions to programs used by many colleagues. When that happens, the end users, who often didn’t expect this situation and are unprepared for it, must suddenly take on the challenges of professional developers, such as testing and maintaining their creations.
This problem offers several interesting research directions for software engineering. When the boundaries between professional and end-user programmers blur, so will the boundaries between the traditional IDE for professionals and end-user tools such as spreadsheets, LabVIEW, or Matlab. Equipping those tools with techniques for testing, measuring, and maintaining artifacts will be an exciting new challenge to help everyone be a programmer, even if for just 15 minutes.

To understand developers in detail, researchers have adopted neuroimaging methods. Their minds have inherent limitations that can be cognitive, such as in terms of working memory or programming skills, or affective, such as tiredness or irritability. To produce high-quality software, developers need support to overcome these limitations. A future challenge of software-engineering research will be to provide such support that’s tailored to the task at hand and takes into account developers’ state of mind.

To this end, researchers need to understand developers in detail. To succeed, researchers have adopted neuroimaging methods to monitor developers’ state of mind. For example, my colleagues and I employed functional magnetic resonance imaging to observe developers as they comprehended source code.1 Igor Crk and Timothy Kluthe used electroencephalography to quantify programmer expertise,2 and Takao Nakagawa and his colleagues used functional near-infrared spectroscopy to assess developers’ cognitive loads.3 Thomas Fritz and his colleagues combined psychophysiological measures to predict perceived task difficulty for developers.4 Measurement devices will become cheaper, smaller, and wearable, such that developers can wear them for their everyday tasks, similar to the activity trackers many of us wear today. So, we’ll be able to tell, for instance, when

- a developer’s cognitive load reaches a critical level and he or she needs extra support,
- a developer is in a state of high concentration and shouldn’t be disturbed, or
- a developer is tired and shouldn’t work on safety-critical tasks.

With such monitoring, IDEs can support developers by, for example, displaying or hiding additional information.

References

Janet Siegmund is a postdoc in human factors in software engineering at the University of Passau. Contact her at siegmunj@fim.uni-passau.de.

FELIENNE HERMANS is an assistant professor of software engineering at the Delft University of Technology. Contact her at f.f.j.hermans@tudelft.nl.

Tailoring Tool Support to Developers’ Needs
Janet Siegmund

SOFTWARE IS DESIGNED, implemented, and maintained by human
Biometric Sensors Will Boost Developer Productivity
Thomas Fritz

PRODUCING GREAT SOFTWARE
as fast as the market demands requires great, productive developers. Yet, what does it mean for developers to be productive, and how can we best help them be productive? To answer these questions, software engineering researchers have been and still are looking predominantly at software developers’ output, such as applications, source code, or test cases. This focus on outputs misses an essential part of software development: the individual developer.

Biometric (psychophysiological) data offers the opportunity to better understand the individual developer and what he or she experiences while working. In particular, biometric sensors let us measure in real time a developer’s physiological features that can be linked to his or her cognitive and emotional states. For instance, changes in pupil diameter sensed through an eye tracker can tell us something about the mental workload a change task imposes.

My vision is to integrate biometric sensing into a developer’s work and leverage the data to boost productivity and provide better, real-time support—for instance, by reducing the number of interruptions at inopportune moments or preventing errors from entering the code. Currently, researchers are only beginning to analyze and understand how to interpret and use such data. Many challenges remain, such as sensor invasiveness, the privacy concerns arising from collecting and using such personalized data, or the real-time cleaning and analysis of fine-grained biometric data. Yet, measuring aspects of an individual developer rather than his or her output will provide a tremendous opportunity to amplify the human in the process and revolutionize software development.

THOMAS FRITZ is an assistant professor in the University of Zurich’s Department of Informatics. Contact him at fritz@ifi.uzh.ch.

Mining Unstructured Data
Gabriele Bavota

UNSTRUCTURED DATA REFER to information that isn’t organized or stored according to a precise schema or structure. Mining unstructured data (MUD) has become popular in software engineering, owing mainly to the high amount of such data in software repositories (for example, issue trackers’ discussions and code comments) and, more generally, on the Web (for example, question-and-answer websites). Although researchers have already exploited this data to provide such support that’s tailored to the task at hand and takes into account the limitations. A future challenge of MUD to support an incredibly large number of software engineering tasks (for example, artifact summarization and bug triage), much more can be done.

First, when talking about MUD in software engineering, people tend to think of textual information spread among software repositories. However, unstructured data are also, by definition, multimedia content, such as images and videos. Our community almost ignores these data, but they embed information that complements what’s available in textual data. For example, a video tutorial can show how a developer interacts with an IDE, something difficult to embed in a textual tutorial. Mining such content could open whole new research directions.

Second, MUD techniques are often applied out of the box to support software engineering tasks. For instance, researchers have used text summarization to provide an overview of the main responsibilities in a class. However, the generated summaries aren’t customized on the basis of who will read them and which task to support. A newcomer in charge of writing the class documentation might need information that differs from what an experienced developer testing the class would need. So, summaries that are consumer-related (who will consume them) and task-related (what they’ll be used for) could lead to the generation of more useful pieces of knowledge. Such summarization is just one of the MUD applications that could be rethought in a consumer- or task-related fashion.

What else to expect? Given the growing amount of unstructured data out there, definitely a lot!
Mining Mobile-App Markets
Meiyappan Nagappan

MOBILE APPS MIGHT differ considerably from traditional software owing to the varieties of mobile devices. This difference might be why interest in mobile apps has increased among both academic and industrial researchers. Academic researchers have shown continued interest in solving classic software engineering problems such as test input generation, software quality, and software reuse for mobile apps. The next logical step is to broaden these solutions to work on cross-platform apps.

On the industrial side, data analytics companies provide mobile-app developers with tools for collecting and analyzing both runtime and user data. These tools let developers know when and how users are using or not using an app. Knowing this, developers can focus on the features that bring in the most revenue.

Another key difference between mobile apps and traditional software is that mobile apps are distributed through centralized app markets that make it easy for developers to release mobile apps. So, the next big idea in software engineering research for mobile apps is mining mobile-app markets.

Through mobile-app markets, researchers have centralized access to not only the apps themselves but also the metadata surrounding them (such as release notes and the security permissions needed) and user feedback (through user reviews and ratings). Some advances in this area have already occurred—for example, understanding the relationship between ratings and downloads and understanding how ads affect both developers and users. Yet, numerous challenges (including mining data continuously in an evolving market and storing and sharing the data) and opportunities (including determining what features to add or how to make an app more competitive) remain to be explored.

MEIYAPPAN (MEI) NAGAPPAN is an assistant professor in the Rochester Institute of Technology’s Department of Software Engineering. Contact him at mei@se.rit.edu.

Energy-Aware Software Development
Abram Hindle

THE FUTURE OF software engineering is energy-aware software development. Software sustainability will become a buzzword. Managers will request that their developers address software energy concerns. Programmers, now responsible for sustainable development, will rely on cloud energy measurement services. IDEs will warn programmers of any dangers to software energy consumption that their changes pose to their product. Their product’s energy consumption profile will be tracked over time to enable careful regression testing and analysis. Third parties will measure, rank, and certify the end product according to “green star”: Software Application Energy Consumption Ratings (SAECRs), an Energy Star derivative for ranking software energy consumption.1

Users will study app energy ratings, rank apps by their energy efficiency, and use such rankings to make app-purchasing decisions. Researchers will employ methodologies (green mining) that carefully evaluate energy consumption by measuring multiple versions of the system under test. This will aim to improve generality and avoid erroneous attribution of energy consumption to nonrepresentative code. Researchers will collaborate on a large shared corpus of software energy runs, enabling development of both general and specific models. These models will be employed by energy-aware plugins in IDEs.

So, the future of software engineering will be energy aware among all stakeholders: users, programmers, managers, and product owners.

Reference

ABRAM HINDLE is an assistant professor in the University of Alberta’s Department of Computing Sciences. Contact him at hindle1@ualberta.ca.

Software Quality Assurance 2.0: Proactive, Practical, and Relevant
Yasutaka Kamei

FUMIO AKIYAMA FIRST attempted to find the relationship between

---

GABRIELE BAVOTA is an assistant professor at the Free University of Bozen-Bolzano, Faculty of Computer Science, Software Engineering Research Group (SERG). Contact him at gabriele.bavota@unibz.it.
size-based metrics and the number of bugs by analyzing actual development histories in 1971. Since then, researchers have devised many ways to help developers with software quality assurance (SQA). For example, my colleagues and I have proposed techniques that automatically identify software changes that have a high risk of inducing defects, by mining the risk that prior changes caused. I see SQA research evolving in at least three directions.

First, SQA approaches must be more proactive. Many of them have been reactive—they determine only what will happen after release. We can devise tools that not only predict risky areas but also generate tests (and possibly fixes) for them. We can also devise techniques that warn developers, even before they modify the code, that they’re working with risky code that has had specific types of defects in the past.

Second, we should evaluate our SQA techniques in practical settings instead of using only traditional measures such as precision and recall. Many SQA studies show that their approaches are 5 percent better than a baseline in terms of precision and recall. However, what does such precision mean for developers? Future SQA research needs to show what that precision means and whether it’s practically effective.

Finally, we should tackle the challenge that new markets raise. One such market is mobile apps. We use smartphones every day and update apps from online stores (such as Google Play). Mobile apps play a significant role in our daily life, and their characteristics differ from those of conventional applications studied in the past. My colleagues and I anticipate new markets to be an area of significant growth.

Reference

YASUTAKA KAMEI is an associate professor in Kyushu University’s Graduate School and Faculty of Information Science and Electrical Engineering. Contact him at kamei@ait.kyushu-u.ac.jp.

Software Engineering for the Web
Ali Mesbah

THE WEB PROVIDES a unique software engineering platform. Its benefits include instantaneous upgrades for all users and “write once, run anywhere” through universal access and execution from any Internet-connected device. Because of these benefits, any application that can be written as a Web application eventually will be. Web-based services such as GitHub and Stack Overflow have already revolutionized how developers write software. This trend will likely continue.

The Web is still in its infancy and is continuously and rapidly evolving. Unlike traditional software, Web apps are heterogeneous (JavaScript, CSS, HTML, and so on). The dynamic interdependencies between these languages, and their distributed asynchronous client-server nature, pose many challenges for developers. This is where software engineering research can play an important role.

As Web languages such as JavaScript become more prominent, IDE support becomes essential for developing and maintaining large-scale applications in practice. However, current software analysis techniques have serious limitations in helping developers understand, write, analyze, and maintain Web code. Current research on Web app code smell detection, refactoring support, and code completion is scarce, and industrial tools available to Web developers have limited capabilities. As it stands today, we’re not even able to extract proper control-flow and call graphs from Web code.

A promising research direction is the ability to handle multiple languages in Web analysis. Recent examples of such research include inferring cross-language slicing and detecting inconsistencies between JavaScript and DOM (Document Object Model). We need interlanguage analyses that can handle Web application code. Also, hybrid approaches that combine static and dynamic techniques will probably prove more useful for this domain.
INVITED CONTENT

References

ALI MESBAH is an assistant professor in the University of British Columbia’s Electrical and Computer Engineering department. Contact him at amesbah@ece.ubc.ca.

Roll Your Own Release (RYOR)
Bram Adams

RELEASE ENGINEERING involves

- integrating developers’ individual code changes into a coherent whole,
- automatically building and testing these changes, and
- deploying and releasing official releases to their target audience.

Over the last five years, release engineering has been revolutionized by a move toward ever shorter cycle times. Major companies such as Google, Facebook, and Mozilla have been reducing the time between consecutive releases from months to weeks, days, or even hours. Such short, regular releases bring features to users faster, gathering feedback more quickly.

Whereas companies traditionally have relied on a central team of release engineers and their automation to coordinate release-engineering activities, companies increasingly are moving toward a roll-your-own release (RYOR) strategy. For example, companies such as Netflix give their developers the power (tools) to automatically build and test a new feature in an environment similar to the production environment. The developers can also, on their own initiative, perform canary releases to certain groups of users. RYOR is a necessity to fully enable new paradigms such as microservices, which decompose a system into many small, independent (and interdependent) Web services.

Now, where’s the catch? Well, traditional release engineers play a unique role. They have a global view of a system’s architecture and quality and have a knack for identifying risky changes that could jeopardize an upcoming release. On the other hand, developers are biased toward their own features. To neutralize this natural bias, RYOR requires help from researchers—for example, to

- automatically predict integration conflicts and the effort to resolve them,
- identify backward compatibility problems,
- profile and optimize the build system and tests, or
- determine the best deployment and release strategy for a certain feature.

To top things off, developers need this support ... inside their IDE!

BRAM ADAMS is an assistant professor at Polytechnique Montréal, where he heads the Lab on Maintenance, Construction, and Intelligence of Software. Contact him at bram.adams@polymtl.ca.

Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.
Focus on Your Job Search

IEEE Computer Society Jobs helps you easily find a new job in IT, software development, computer engineering, research, programming, architecture, cloud computing, consulting, databases, and many other computer-related areas.

New feature: Find jobs recommending or requiring the IEEE CS CSDA or CSDP certifications!

Visit www.computer.org/jobs to search technical job openings, plus internships, from employers worldwide.

http://www.computer.org/jobs

The IEEE Computer Society is a partner in the AIP Career Network, a collection of online job sites for scientists, engineers, and computing professionals. Other partners include Physics Today, the American Association of Physicists in Medicine (AAPM), American Association of Physics Teachers (AAPT), American Physical Society (APS), AVS Science and Technology, and the Society of Physics Students (SPS) and Sigma Pi Sigma.
Imagine you’re the head engineer on a team of dedicated software developers. You’ve all been working hard for years to bring that new piece of software to the market—developing bleeding-edge algorithms, optimizing all parameters, paying for licenses, and more. And, of course, all your infrastructure is secure: nothing goes in or comes out without scrutiny. However, at some point, you must provide your software to customers. What’s to prevent curious customers (some of whom might be your competitors) from peering into your program’s internals, reverse-engineering its functionality, and selling their own version?

Software obfuscation might help avoid such situations by producing a program that, even if its code could be seen, its function would be utterly incomprehensible. By “incomprehensible” we mean that the computer could understand and execute the code, but engineers—even those assisted by automated tools—would find it extremely challenging to comprehend.

For a long time, software obfuscation was a matter of taste and experiment, with little theory to support its practice or measure its efficiency. Now, with advancements in cryptographic research, formal and secure code obfuscation is more achievable than ever before.

A major cryptographic breakthrough in 2013 led to a candidate for a cryptographic construction: indistinguishability obfuscation. For the first time, a potential formal basis for code obfuscation was within reach. This sparked a spate of publications on the topic. However, despite the excitement, researchers encountered many technical challenges; for example, practical implementation of the proposed obfuscation was excruciatingly slow and not nearly as secure as hoped. But these papers generated a good starting point, community-wide excitement, and new determination motivated as much by overcoming scientific challenges as by the need for program obfuscation.

From Spoof to Proof
We can think of code obfuscation as a procedure by which a program is made unintelligible. This is done in such a way that the functionality of the program won’t be affected, but it will be very difficult to reverse-engineer how the program works.

For example, the following JavaScript program prints “I love you”:

```
[ [ ] + 1 / ! 1 ] [ 1 ^ 1 ]
[ 1 > 1 ] + ( () + [ ] )
[ 1 < 1 ^ 1 ] + ( [ ] + ! ! - [ ] ) [ 1 << 1 ] + [ / ~ / + ( ) ]
[ + ! 1 ] [ - ~ 1 << 1 ] + / \ [ ^ 1 ] + \ ] / [ [ ] + ! [ ] ]
[ 1 << 1 ] + ( / | \ [ 1 + { } ]
[ 1 + 1 ] + ( [ ] + {} ] + [ ] + [ ] )
```

CDOE OBOFSUCAITN:
Securing Software from Within

Marc Beunardeau, Aisling Connolly, Rémi Géraud, and David Naccache | École Normale Supérieure
This code works by generating a series of messages and extracting the required letters from them. For example, to extract the “I” at the beginning of the phrase, “Infinity,” [1^1] evaluates to zero, which lets us position ourselves at the start of the array. Finally, [1>>1] returns the character at that position. Altogether, [1<<1][1^1][1>>1] yields “I.”

This is a nice obfuscation: it satisfies our criteria of being unintelligible, a machine can read it to produce the desired output, and it remains difficult to reverse-engineer. However, these techniques work well because they exploit traits of the specific programming language (and specific browsers). Can we build an obfuscator that’s independent of language, OS, or browser? Furthermore, can we build a universal obfuscator that can ensure that our programs are securely unintelligible?

**Black-Box Obfuscation**

Despite code obfuscation’s usefulness in various settings, it was not until the turn of the 21st century that it received rigorous cryptographic treatment by, among others, Satoshi Hada and Boaz Barak and his colleagues.

The first intuition of obfuscation we described led to what specialists dubbed black-box obfuscation. With black-box obfuscation, an adversary shouldn’t be able to learn anything about how a program works. But Barak and his colleagues showed that this isn’t fully achievable: there are always “unobfuscatable” codes.

For instance, consider a quine, a program that outputs its own source code. It’s a standard exercise for computer science students to write such programs. No matter how obfuscated a quine will always output a clean version of its source code. What’s worse, we can’t “just remove” problematic programs like quines from the programs we want to obfuscate: Barak and his colleagues further showed that there are unobfuscatable “function families” and that these problematic programs are ubiquitous.

**Indistinguishability Obfuscation**

Barak and his colleagues’ impossibility result was based on the assumption that we’re seeking black-box security. But what if this security assumption was weakened a little? The authors proposed (in the appendix of their celebrated paper) a slightly weaker yet “best-possible” notion of obfuscation: indistinguishability obfuscation (iO).

Very briefly, iO is achieved when you don’t know which of two programs $C_0$ or $C_1$ has been run. Of course, this requires that $C_0$ and $C_1$ behave very similarly.

We denote the programs as $C_0$ or $C_1$ because in this setting, for ease of computation, we consider them to be circuits. In this case, a circuit is simply a series of AND, OR, and NOT gates, which, if combined in the right way, allow us to construct any program. To be more precise, to achieve iO, we require that given any two functionally equivalent programs $C_0$ and $C_1$, represented as circuits of similar size, the obfuscations of $C_0$ and $C_1$ should be computationally indistinguishable. For example, let $C_0 = a(b + c)$ and $C_1 = ab + ac$. In this case, $C_0$ and $C_1$ are functionally equivalent. If $C_0 = C_1$ for all $a$, $b$, and $c$, then we can say that $O(C_0) \approx O(C_1)$; that is, the obfuscations of these programs are computationally indistinguishable.

In practice, iO is sufficient for most applications, although it doesn’t quite satisfy obfuscation’s intuitive concept. Among known iO applications are public-key encryption from symmetric-key primitives, witness encryption, deniable encryption, functional encryption, multiparty key exchange, efficient traitor tracing, and so on. Anant Sahai and Brent Waters outline a host of useful recipes.

**Indistinguishability Obfuscation from Homomorphic Encryption**

Let’s briefly review how homomorphic encryption works. (For a deeper overview, we refer readers to the work of Frederik Armknecht and his colleagues, as well as to Marc Beunardeau and his colleagues’ earlier column in *IEEE Security & Privacy*.)

A homomorphic encryption scheme is a classical public-key system with an additional “homomorphism” property. This property lets us perform an operation on two encrypted messages so that the result is a valid encryption of a new message. The typical example is that of (unpadded) RSA. From
ciphertexts $c_1$ and $c_2$, we can compute $c_3 = c_1 \times c_2$, which is a valid ciphertext. When decrypted, $c_3$ yields the plaintext $m_3 = m_1 \times m_2$.

In most cases, there’s only one homomorphic operation, usually either addition or multiplication. But two well-chosen operations (say multiplication and addition) are enough to perform arbitrary operations if we combine them appropriately. Constructing such systems isn’t an easy task, but several homomorphic schemes are available for use. In 2009, Craig Gentry proposed a first candidate: a fully homomorphic scheme that has since been built upon in useful ways. For example, Microsoft recently published its “Manual for Using Homomorphic Encryption for Bioinformatics.”

That being said, how can homomorphic encryption provide an elegant way to implement indistinguishability obfuscation? Let’s compare homomorphic encryption and indistinguishability obfuscation (see Figure 1).

There’s an attractive symmetry between the two schemes, one we can make explicit by turning a homomorphic encryption scheme into an obfuscator. The steps are as follows:

- Provide an encrypted description $C$ of a program $P$. This is the obfuscated code. Because we use fully fledged encryption, this program is cryptographically secure. Recovering $P$ from $C$ would require breaking the encryption scheme.
- Provide an encryption $I$ of some input, for instance, an encryption of 0 or 1. Anyone can encrypt, so any user can provide the encrypted input.
- Run the public, homomorphic operation of “evaluation”: $eval(C, I)$. This operation is performed directly on the encrypted program $C$ and encrypted output $I$. But thanks to homomorphic encryption, it results in an (encrypted) output $O$, which is the correct outcome of the computation $P$.

Because $O$ is encrypted, we discover the computation’s output through a final step: adding a “zero test” procedure. This procedure enables anyone to know whether an encrypted message ($O$ in our case) is zero.

Note that incorporating zero tests isn’t innocent; in practice, many proposed constructions were broken directly as a result of abusing zero tests.

**iO’s Status**

iO is deeply linked to other cryptographic primitives. In fact, its applications are far more diverse than hiding software code from competitors. For example, iO enables us to turn a symmetric cryptosystem into an asymmetric one by making the encryption algorithm public after obfuscating it. Anyone can use it to encrypt, but the key contained in the program is obfuscated and thus can’t be recovered by an adversary.

From a more theoretical angle, iO yields the existence of one-way functions. One-way functions—functions that one can’t efficiently invert—are essential to almost all cryptography fields. For this reason, many works rely on the hypothesis that one-way functions exist or that some primitives behave like one-way functions. We can say that iO implies the existence of these one-way functions precisely because of iO’s security definition. Under iO, programs are computationally indistinguishable. Therefore, we can say for certain that no efficient distinguisher can determine from which circuit the output was produced; that is, there’s no efficient inversion.

Another cryptographic problem that’s been open for 16 years—deniable encryption—could be solved by proving iO’s existence. Deniable encryption provides plausible deniability in case an adversary accesses your key or randomness source.

But why are we talking so cautiously about iO’s existence? Because, alas, all we have so far are candidate constructions. Ultimately, the problem lies in combining homomorphic encryption with zero tests in a device called a multilinear map. The first multilinear maps were invented in 2013. In 2015, two waves of cunning cryptanalytic attacks showed that they were insecure, enabling researchers to recover secrets, first by abusing the zero tests, and later even without them.

**Table 1.** Comparison of homomorphic encryption and indistinguishability obfuscation. The symmetry between the two schemes enables us to leverage homomorphic encryption to build an obfuscator.

<table>
<thead>
<tr>
<th>Homomorphic encryption</th>
<th>Indistinguishability obfuscation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Take encrypted input</td>
<td>Take clear input</td>
</tr>
<tr>
<td>Perform public operation</td>
<td>Perform “obfuscated” operation</td>
</tr>
<tr>
<td>Output encrypted message</td>
<td>Output clear result</td>
</tr>
</tbody>
</table>

**Figure 1.**
can be made rigorous, and the plans to achieve this are laid out. What remains is for a generation of computer scientists and engineers to acknowledge and further these insights, bringing better security and privacy to everyone’s software.

References

Marc Beunardeau is a PhD candidate at the École Normale Supérieure. Contact him at marc.beunardeau@ingenico.com.

Aisling Connolly is a PhD candidate at the École Normale Supérieure. Contact her at aisling.connolly@ens.fr.

Rémi Géraud is a PhD candidate at the École Normale Supérieure. Contact him at remi.geraud@ens.fr.

David Naccache is a professor of computer science at the École Normale Supérieure. Contact him at david.naccache@ens.fr.

Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.
WE ALL KNOW THAT CLOUD COMPUTING IS EXPLODING AND WE HAVE THE ANALYST NUMBERS TO PROVE IT. Indeed, the most interesting data points in a recent Gigaom (gigaom.com) survey are the number of instances currently running within enterprises. The survey results suggest a pattern of public cloud acceptance and adoption, with more than 50 percent of enterprises surveyed running between one and 50 public cloud instances on a typical day, another 32 percent running more than 51, and 4 percent—cloud “super users”—running more than 1,000 cloud instances in any given day. What’s most interesting about the survey results is that the number of instances reflecting the true use of the cloud is rather impressive relative to what we’ve seen in the past. Enterprises are putting more systems and applications into production on public clouds, and thus spinning up more instances to support an increasing processing and storage load.

That said, most existing independent software vendors (ISVs) don’t have a true cloud offering (typically, an application, API, or data, but it could be any type of service offered by a cloud service provider). Indeed, they’ve been struggling in a market where software solutions that are delivered using the public cloud model have a clear leg up on those that don’t. Therefore, ISVs have been working to create cloud-based versions of their software. However, finding the right path to getting to the cloud can be difficult.

However, not all ISVs should be in the cloud, or it might not be economical for them to move. About 25 percent of existing ISVs have software that doesn’t have a platform analog in a public cloud. Thus, they have to build their own cloud, which is hugely expensive, or they have to wait for a public cloud service equivalent to appear on the market. An example is ISVs based on mainframe platforms.

These ISVs face some tough choices. First, does the cost of migration justify the value delivered? Second, will they be able to deliver a working system that will meet the needs of their existing user base? Finally, can they capture a new market with their cloud offering, and how big will that market be?

What’s an ISV to Do?
How can ISVs take advantage of the changing market? The kneejerk reaction is to push their offerings to the cloud, no matter if they should be there or not. Indeed, 10 years ago when the software-as-a-service (SaaS) explosion occurred, many ISVs declared themselves SaaS providers. Although some did okay within the shifting tides of software consumption, many didn’t, and many actually went away.

The patterns of SaaS enablement were almost humorous. Some ISVs would provide a dedicated server for each user on their SaaS cloud. This means that when they sold a cloud service, they would have someone go to the datacenter and install a new server in a rack. Or, better yet, assign a server from a pool of servers already installed.

This primitive and expensive approach to multititenancy was meant to be just a stopgap until they could get a true cloud service into the market. However, users figured out the approach quickly, and many of those ISVs lost credibility in the emerging cloud market.
Today we have thousands of ISVs that haven’t made the journey to the cloud yet. Or, they might have attempted the journey in the past, but failed quickly. These still-thriving businesses, many of which were waiting to see if the cloud would fade away, are now facets with the fact that cloud computing is here to stay, and they need to find a way to adapt. Otherwise, they’ll watch the market go away.

Several key issues are driving the need to migrate to the cloud.

First, there’s pressure on ISVs to meet the SaaS requirement. Although budgets are coming back, slightly, chief information officers (CIOs) are being asked to do more with much less. Thus, the notion of paying another hundred thousand to many millions of dollars in software license costs is, to many enterprises, out of the question. As one CIO stated, “We want Salesforce.com versions of everything.”

In addition, there’s a movement away from capital expenditures. Although the value of cloud computing around operations expenses versus capital expenditures is well known, it’s being pounded home with the number of enterprise datacenter projects that are now awaiting funding. Management in those companies are now questioning the need to build or rent more datacenter space, and are pushing back on CIOs who are looking for cloud-based applications to replace many local applications.

Finally, we’re seeing the demise of the cloud computing “Chicken Littles.” A Gigaom study on cloud adoption rates found that 80 percent of enterprises surveyed had migrated at least some of their non-mission-critical assets to the cloud, with 5 percent claiming to be “all in.” This level of maturity indicates that we’ve moved from the experimental and skeptical state to acceptance of cloud computing as a sound direction. Those who pushed back on cloud computing because of security and reliability issues haven’t seen these issues arise as real concerns, and most enterprises are well into their second or third implementations.

Thus, not only is the market asking ISVs to redo their software as cloud services, but their existing customers are doing the same.

Most ISVs have been looking at cloud enablement over the years, and perhaps have even invested in some pilot projects, only to balk at the cost and the risks around building a true cloud-based offering. Some ISVs never got beyond single-tenancy offerings, which were just too costly to put into production (as noted earlier).

As Figure 1 illustrates, approaches to application development and deployment are changing around the use of cloud computing. In the “old world,” traditional architectures and approaches to application development and deployment had limitations, such as the inability to leverage distributed platforms because the application was too tightly coupled with the database. The “new world” offers more advantages, but only after an organization has dedicated a significant amount into careful planning, architecture, and technology selection.

It’s important to note that this is perhaps more important than the move to the cloud itself. The ability to build automated processes, such as self-healing, into cloud-based application offerings and the use of DevOps is where the ISVs will see value. Additionally, they’ll see value in the ability to scale up and scale back in direct proportion to the revenue generated, as well as moving to more modern development approaches based on use, services, or APIs.

**Why to Cloud**

ISVs have to do some work on the front end to determine the actual value of moving. They need to figure out whether

![FIGURE 1. Changing approaches to application development and deployment. (Source: Cloud Technology Partners)](https://example.com/figure1)
they’ll benefit from migrating to the cloud before they toss money at the issue. For various reasons, not every software system should exist in the cloud. However, most should, and that should at least send ISVs to do some quick research as to the potential value that a cloud-based offering can bring, or perhaps how to avoid bankruptcy-by-cloud.

ISVs should consider the following advantages of cloud adoption:

- competitive advantage,
- ability to rapidly adapt to new market opportunities,
- speed and agility to deliver new functionality and features,
- dynamic global scalability,
- opportunity to capture the long-tail of market (cost of delivering application on traditional infrastructure is too high for small and medium-sized businesses),
- lower total cost of ownership (to avoid low utilization of over-provisioned existing infrastructure), and
- faster customer onboarding.

Benefits to ISV customers include

- more flexible pricing and delivery models,
- configurable services (one size does not fit all), and
- lower maintenance effort and cost.

ISVs moving to SaaS solutions, or perhaps just a selection of cloud services or APIs, must do a great deal of work to create competitive public cloud-based offerings. They must be able to deal with tenancy issues as well as provide Web-scale features, such as

- usage-based accounting,
- client management,
- data integration and migration solutions, and
- cloud native performance.

The path to get to these “cloudy” features can be rather simple for software systems that have been well designed, maintained, and implemented. However, ISVs that have been around more than a few years are typically dealing with older legacy code and databases, or layer upon layer of code on top of code that was supposed to be fixed, but never was. Thus, getting to the cloud could require a great deal of rewriting and re-architecting, perhaps leading to more risk and cost than any value cloud enablement could bring.

Thus, the core question in the conference rooms of most older software players isn’t if they can cloud-enable their systems—they certainly can with enough money. The question is whether there will be value once they take the risk and spend the money to move their software to the cloud, providing a true SaaS solution with all of the features the industry, and their customers, will expect.

How to Cloud

After dealing with “should we cloud,” it’s time to address “how we cloud.” Figure 2 depicts one approach to cloud enablement, working from the initial vision of the cloud solution to its operation. This includes defining the target market for the SaaS cloud, which leads to the core requirements. From there, ISVs need to define any changes to the architecture and enabling technology, then move to development, testing, and deployment, typically leveraging agile and DevOps best practices and technologies. Next, they need to go to market, defining all aspects of operations around the new cloud service or services.

As a general rule, organizations will spend about 10 to 20 percent of their yearly software revenue on moving a software system to a cloud-based solution. Thus, if a company makes $0.5 billion, it’ll spend about $50 to $100 million on the refactoring, re-architecture, and re-development to get to the right cloud offering. Of course, the normal consulting answer of “it depends” comes to mind.

**FIGURE 2.** An example cloud-enablement process, from initial vision to operation. (Source: Cloud Technology Partners)
in that some ISVs will find they need much more time and money to cloud-enable their software, whereas some that have done a better job on architecture and coding can move rapidly to the cloud, perhaps in less than six months.

Therefore, you should do some sound self-assessment and planning. These tasks typically include determining the “as is” state of your software, such as how things were coded and the overall architecture. If you find limiting issues (for example, the database is too tightly coupled to the application), you need to correct them, which will take more time and money.

Another task is to create a logical “to be” architecture that maps to a “to be” physical architecture. This means that you’ve thought through how to update the application, if needed, and have determined how to map the application to the new physical cloud platform. Items often overlooked include security, governance, and database efficiency. These need to be addressed in terms of the solution and enabling technologies employed.

Finally, make sure to build a DevOps process including tools. The ability to continuously improve the cloud-based solution using DevOps approaches and technologies will provide more value to your users and your business. There should be well-automated development, testing, and deployment.

IF YOU’RE AN ISV THAT HASN’T MOVED TO THE CLOUD OR PERHAPS MOVED TO THE CLOUD AND FAILED, YOU UNDERSTAND THE IMPORTANCE OF GETTING YOUR CLOUD IMPLEMENTATION AND DEPLOYMENT RIGHT—THE FIRST TIME. To be successful, you need to determine where you are and where to begin, as well as the architectural paths you need to take, the processes you need to change, the development practices you should augment, and finally, how to make your software operationally successful in the cloud.

DAVID S. LINTHICUM is senior vice president of Cloud Technology Partners. He’s also Gigaom’s research analyst and frequently writes for InfoWorld on deep technology subjects. His research interests include complex distributed systems, including cloud computing, data integration, service-oriented architecture, and big data systems. Contact him at david.linthicum@cloudtp.com

Gambling with Source Code

Nir Kshetri, University of North Carolina at Greensboro

Tech companies required to comply with controversial regulations in foreign countries as a cost of doing business there must trade off the potential economic benefits with the risk of long-term harm.

In October 2015, IBM agreed to let officials from China’s Ministry of Industry and Information Technology (MIIT) examine the source code of some of the company’s products, which are widely used by Chinese firms in the financial, Internet, and energy sectors—supposedly to ensure that it had no “security flaws.” IBM justified its controversial decision to comply with the demand, which took the industry as well as the Obama administration by surprise, on the need to expand market share in China and cited earlier agreements by Microsoft to allow the Chinese as well as Russian and UK governments to access Windows source code.1,2

However, critics argue that IBM is putting its intellectual property (IP) at risk, and that its action could encourage other US tech companies seeking a competitive advantage overseas to likewise reveal their “secret sauce” to foreign governments.3 While providing short-term economic benefits, this could ultimately damage companies’ long-term prospects as well as compromise both their own and national cybersecurity infrastructure.

Chinese computer users.1 Due to Chinese policymakers’ deep-rooted suspicions about spying by Western governments and corporations, computer hardware and software imported from the US and its allies are subject to detailed inspection. Analysts argue that China’s broader goal is to create a domestic information and communications technology (ICT) industry that will minimize reliance on foreign ICT and its associated security dangers.3

As far back as 2000, newspaper reports highlighted China’s long-standing distrust of Western ICT and its impact on national security.4 For example, an 8 February editorial in the People’s Liberation Army Daily noted that “some countries” with highly developed ICT industries are “taking advantage of their monopolistic position” to “control information technologies, infiltrate information resources,” and dump ICT products in underdeveloped countries “to attain political, economic and military objectives.” In the China Economic Times of 12 June, Xu Guanhua, then Chinese vice minister of science and technology, argued that a key aspect of developed countries’
military strategy was exploiting exported software for “coercing, attacking, or sabotage.”

Revelations in 2013 by Edward Snowden, a former CIA employee and NSA contractor, of mass US government surveillance heightened Chinese fears. Commenting on US intelligence agencies’ alleged hacking of China’s major mobile companies and universities, an editorial by the Xinhua News Agency noted: “These, along with previous allegations, are clearly troubling signs. They demonstrate that the United States, which has long been trying to play innocent as a victim of cyber-attacks, has turned out to be the biggest villain in our age.”

Following the US government’s indictment of five high-ranking Chinese army officials in May 2014 for alleged cybercrimes against US companies, China announced that it would perform rigorous national security inspections of imported tech products. Nationalist bloggers described the inspections as a “hard blow to anti-China forces” and suggested that US companies such as Cisco, IBM, and Microsoft would be negatively affected.

A NEW REGULATORY ENVIRONMENT IN CHINA

The first sign of trouble came in January 2015, when China required foreign companies selling ICT to its banks to turn over source code, submit to government audits at any time, and build back doors into their products that the Chinese government could access. US and other Western firms, with their governments’ backing, strongly objected to the new policies and complained that they amounted to protectionism.

China suspended the requirements in April 2015, but some analysts correctly predicted that China would try to achieve the same end in a different way. In July 2015, China’s parliament, the National People’s Congress, drafted a new law calling on the Chinese government to define national and industrial cybersecurity standards that all technology vendors must comply with. Critics suspect that such standards could exclude foreign vendors or require them to provide the Chinese government or domestic firms with access to their IP.

China’s real, unstated goal appears to be to remove most foreign technologies from its banks, military systems, state-owned enterprises, and key government agencies in the near future. For example, the China Banking Regulatory Commission requires banks and financial companies to have at least 75 percent of their computer systems using “safe” technology by 2019 and to spend at least 5 percent of their ICT budgets for this purpose.

THE CARROT AND THE STICK

Foreign businesses operating in China deemed friendly to the government have easier access to China’s enormous but tightly regulated market. However, noncompliance with government regulations and/or harsh criticism of the regime can lead to severe restrictions and even exclusion from that market. For instance, foreign media outlets, blogs, and social media sites that try to flout China’s strict Internet censorship and surveillance laws are often blocked.

China is actively striving to reduce its dependence on traditional manufacturing and diversify its economy. In 2015, President Xi Jinping laid out plans to accelerate development of a native ICT industry. To nurture this industry’s growth, the government relies on a system of subtle rewards and punishments to acquire needed technologies from foreign firms.

In March 2015, IBM chairman, president, and CEO Virginia M. Rometty noted that foreign companies

To nurture growth of its domestic ICT industry, the Chinese government relies on a system of subtle rewards and punishments to acquire needed technologies from foreign firms.

THE RISKS OF COMPLIANCE

What risks is IBM taking by revealing its products’ source code to Chinese officials? Is it possible for them to copy or steal it and pass it on to Chinese companies developing competing products? If so, how much will this impact IBM?

IBM’s source code demos reportedly will last only a few hours and be performed in a room without an Internet connection. These restrictions make it extremely difficult for government officials involved in the demo to steal the code.

It’s also important to point out that software development is highly skill
intensive and requires more tacit than explicit knowledge, generally making it more difficult to imitate and reproduce software than hardware. In addition, unlike hardware, software is evolutionary in nature, requiring continual updates and modifications. Even if you acquire a product line’s source code, the code is likely to change so quickly as to become obsolescent by the time you’re in a position to launch a clone product. This characteristic is akin to the high velocity of big data—most data has a short shelf-life, with its value declining exponentially over time.

Moreover, a key strength of tech giants like IBM is the ability to provide integrated solutions for customers. IBM isn’t a specialized manufacturer, so even if Chinese companies are able to develop some products or services using stolen code they’re unlikely to significantly threaten IBM’s overall business operations. The risks of compliance, then, are real but can be overstated.

**IBM’S DILEMMA**

US government officials and companies are equally distrustful of their Chinese counterparts. In a 2011 Wall Street Journal editorial, former White House national security official Richard Clarke asserted that “Beijing is successfully stealing research and development, software source code, manufacturing know-how and government plans.” Critics claim that the Chinese government routinely sponsors or sanctions cyberattacks on Western government agencies and companies and shares the knowledge obtained from these attacks with Chinese enterprises.

Beyond general national security concerns, US tech companies worry that complying with China’s regulations could open the door to theft of their IP and weaken their cybersecurity infrastructure. They also worry about their public image: because of China’s strict censorship laws and its reputation for punishing outspoken dissidents, businesses that cave to Chinese government demands—particularly those that impact the Internet and social media—are often criticized for putting profit before democratic principles.

The other so-called BRIC countries—Brazil, Russia, and India—might follow China’s lead and demand that IBM and other US tech companies reveal their products’ source code to continue doing business there. Their motive might simply be to ensure that the products are indeed secure given the number of highly publicized cyberattacks on US assets. But given China’s stated desire to create a home-grown IT industry and the long history of mutual suspicion between the US and China, including tit-for-tat cyberwarfare allegations, US companies might want to think twice about handing over source code to Chinese regulators—or at least proceed very cautiously.

To understand why IBM would risk doing so, even under highly controlled conditions, it’s important to appreciate the dilemma the company faces. In June 2015, IBM reported that overall revenue from its products had declined consecutively for 13 quarters; the falloff was particularly acute in China, where revenue had declined by 40 percent during the previous nine months. The Asia-Pacific region, excluding Japan, accounted for 14 percent of the company’s 2013 revenue of $99.8 billion, and China accounts for a significant part of this revenue.

IBM is thus actively pursuing expansion opportunities in the Chinese market. While announcing the source code demos, IBM senior vice president Steve Mills acknowledged that the company needs the Chinese government’s support to grow its business in the country. By revealing its products’ source code, IBM might be hoping to convince officials that it has nothing to hide. The unanswered question is: what will China do with the information?

**REFERENCES**


**ACKNOWLEDGMENTS**

I thank Jeff Voas for numerous edits and suggestions on previous versions of this article.


NIR KSHETRI is a professor of management in the Bryan School of Business and Economics at the University of North Carolina at Greensboro. Contact him at nbkshetri@uncg.edu.

This article originally appeared in Computer, vol. 49, no. 2, 2016.
THE IDYLL has been going on for decades. DevOps, the synergy between software development and IT operations, was an open secret before it became a mass movement. Passionate programmers were often also closet system administrators—sometimes literally so, by nurturing recycled hardware in their home’s closet. These same programmers were also drawn to the machine room, chatting with the administrators about disk-partitioning schemes, backup strategies, and new OS releases. Not to be outdone, zealous administrators would find endless excuses to develop all sorts of nifty software: deployment automation, monitoring, provisioning, and reporting tools.

Many factors are propelling the increased adoption of DevOps. First, software is increasingly being offered over the Internet as a service instead of being developed as an organization’s bespoke system or a shrink-wrapped product. This makes operations an integral part of the offering, driving demands for service quality. Then there’s the agile movement. Its emphasis on cooperation between all stakeholders has helped formalize the relationship between development and operations. Its acceptance of change has driven demand for processes and tools that let systems respond to change swiftly and efficiently. Another enabler has been the availability of powerful and plentiful hardware. It has allowed the abstraction of system infrastructure and its expression as code amenable to established software development practices. Resource virtualization and cloud computing have provided the required building blocks.

In many IT sectors, DevOps is here to stay, helping deliver higher-quality services more efficiently. How can you, as a software practitioner, embrace DevOps to increase your organization’s effectiveness?

// TODO
Start by cooperating more closely with your IT operations colleagues. Involve them in all stages of your software’s development. Elicit their requirements to find which tools and APIs they need to deploy the system efficiently and to manage it effectively in production environments. Exchange views on architecture and features that will make your software more reliable, more scalable, and easier to deploy, configure, and run. See how you can issue software releases that painlessly mesh with running systems. Discuss planned changes and how they’ll affect operations. Many old-style organizations have development and operations work in disjoint silos. Strive to break these down by instituting regular meetings, setting up shared (virtual and physical) workspaces, and embedding people from one group into the other.

If your application domain allows it, let agile-development principles guide your relationship with operations. Prefer to interact with your operations colleagues rather than be guided by rigid processes. Collaborate with them to solve problems rather than fight over service-level agreements. Focus on software that’s running and delivering a service rather than on comprehensive documentation and formal handovers.
Help operations respond to events on the ground rather than hide behind an established plan. Remember: the operations team is one of your software’s users. Framing your nonfunctional requirements as user stories aimed at the operations team makes ops a first-class stakeholder.

Expand the cooperation at the technology level. Start by extending your continuous-integration cycle to include the software’s deployment on a test server. As you gain confidence with this, carry on with planning and practicing continuous delivery: the deployment of each software release to production. Add feature toggles and other infrastructure required to control your customers’ experience and satisfy your business model.

First-class operations teams hate manual work (they call it toil, and they strive to eliminate it), so make your product amenable to automation. Ensure the software’s deployment and runtime behavior can be easily controlled from the command line or through scripting APIs. Document these features and give them first-class status by designing any manual operation methods to work through the automation interfaces. Use file formats that other programs can easily parse and generate. Similarly, adopt control interfaces, such as REST (Representational State Transfer), that can be used with minimal ceremony. You don’t want to bury your operations counterparts under five layers of abstraction and hundreds of dependencies.

First-class operations teams also hate flying blind. Alerts help them respond quickly when problems arise, while trends provide feedback from operations back to development. Equip your software with mechanisms that let others monitor its functionality and performance. Use a full-featured logging library, include logging statements in your software, and document the interfaces that control the logging verbosity. Write logging statements that the operations team can easily dissect, correlate, and aggregate to analyze your software’s operational performance. If your systems support a whole-stack tracing tool, such as DTrace or LLTng, detail its use to scrutinize your software’s operation. Provide ways through which system-monitoring watchdogs, such as Nagios plug-ins, can verify that your software is alive and well. For higher marks, provide information regarding your software’s load and performance metrics, such as throughput, latency, resources used, and uninvoked requests.

Finally, as you cooperate more closely with your operations colleagues, strive to learn from each other. You can help your operations team transplant into their setting your successful development practices, methods, architectures, and tools: how you use revision control tools to develop on multiple branches, how you document useful designs as patterns, how you program in pairs, and how you perform continuous integration. You can also learn a lot from the operations side: the relentless focus on service, quality, and reliability; the control of risk; the organization of complex deployments; the use of system configuration management tools; and the elimination of toil.

thinking like a DevOps developer is an essential trait of an enlightened software professional. 😊
Ingo Weber, Surya Nepal, and Liming Zhu • Data61, CSIRO, and the University of New South Wales

In this article, the authors analyze the security and dependability challenges for developing dependable and secure cloud applications. They also provide an overview of their research and development that aims to ameliorate some of the obstacles.

An increasing number of applications are provided as Software-as-a-Service (SaaS) solutions on the basis of public cloud services, such as Infrastructure-as-a-Service (IaaS) or Platform-as-a-Service (PaaS) offerings. Cloud services and the ability to manage and control them programmatically through APIs enabled the rise of continuous deployment and DevOps. DevOps, the abbreviated combination of “development” and “operations,” has been defined as “a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality.”¹ This methodology is rapidly progressing toward mainstream adoption,² as it delivers new functionality to end users fast and often increases quality along the way.

However, achieving security (confidentiality, integrity, and availability) and dependability (availability, reliability, safety, integrity, and maintainability) when developing applications in the context of cloud services and DevOps offers a separate set of challenges. For instance, DevOps typically includes implementing a continuous deployment pipeline (CDP), which automatically tests and deploys new software versions. This CDP needs to be secured and checked for errors — or else it could spread malicious or erroneous code to all servers, thus easily multiplying any problems by orders of magnitude. Trying to achieve full automation in testing and deployment also puts additional challenges on security and dependability, including requiring better practices around quick recovery, rollback, and resilience.

Here, we broadly discuss such challenges by analyzing the security and dependability concerns for all phases of software development and data security life cycles of SaaS solutions. We also provide an overview of our research and development that aims to alleviate some of the pains of the process.

**SaaS Security Life Cycle (SSLC)**

SaaS follows the standard software development life cycle (SDLC), which includes requirement gathering and analysis, design, implementation or coding, testing, deployment, and maintenance or operation. Technologies like DevOps have significantly reduced the time needed from implementation over testing to deployment, through the use of a CDP and other means. The SDLC uses a number of artifacts (such as design documents and test datasets), including the I/O data used during the operation phase. Hence, it’s important to consider the dependability and security of cloud applications in a joint SaaS security life cycle (SSLC), which includes the SDLC and the data security life cycle (DSLC). The standard DSLC includes the phases of creating, storing, using, sharing, archiving, and destroying. Figure 1 shows the SSLC that combines the SDLC and DSLC. It has four “Development” and “Operation” phases and one ongoing data storage phase that extends across all four of
the other phases. Next, we describe the security and dependability issues in each of those five phases.

**Development**

The development phase in the SSLC includes the following four SDLC steps: requirements gathering, design, implementation or coding, and testing. The first two steps define the security and dependability requirements of the application, whereas the last step includes most activities of the CDP. The implementation and testing steps can be carried out in the spectrum between two extreme ways: fully traditional or fully in-cloud. In the fully traditional approach, the development phase occurs entirely on developers’ local machines or in the organization’s network, and the cloud only comes into play in the deployment phase. In contrast, the fully in-cloud approach uses cloud resources for implementation (development environment, revision control, and coordination) and testing (using cloud-based testing environments and a cloud-based CDP).

Most development teams choose a middle ground of sorts — for example, with local development, revision control in the cloud, local integration testing, and cloud-based acceptance testing. Though in-cloud solutions provide a number of benefits, such as agility and distributed collaborative coding, they also add a number of threats with regard to security and dependability. For each component hosted in the cloud, the same threats apply as for the application itself, so we postpone this discussion until later in the execution and maintenance phase. Furthermore, inability to port developed applications from one cloud provider to another has become a critical issue. The standard techniques such as static application security testing (SAST) and runtime application self-protection (RASP) should be a key part of cloud application development.

**Delivery and Deployment**

This phase includes delivery and deployment of applications to the cloud. The dependability and security of cloud applications can be enhanced by using continuous delivery or continuous deployment. Continuous delivery ensures that all changes in an application are seriously tested and ready to be deployed. Continuous deployment goes a step further and automatically deploys those changes, as long as the quality gates are passed. We developed a solution for dependable deployment of cloud applications, called the Process-Oriented Dependability (POD) framework — see the related sidebar for a brief description. We also worked on analyzing security aspects of the CDP, and what it would take to secure the CDP. Continuous deployment further requires better
security and dependability, such as specific monitoring solutions and fully automated rollback plans so that a botched deployment can be recovered quickly.

**Execution and Operation**

Cloud applications are inherently vulnerable due to the underlying business model, which builds on multitenancy, and the virtualization technology that enables such a model. Multitenancy introduces a new set of dependability and security challenges. The threats to integrity, reliability, and secrecy of both processes and data stem not only from the remote adversaries, but also from local adversaries, such as providers’ administrators or other tenants.

Virtual machines (VMs) and VM images need to be secured throughout their life cycles in all three states: creation, running, and dormant. For example, dormant VMs are often forgotten when performing security upgrades or patches, resulting in increased risk when they’re brought back online. The virtual machine monitor (VMM), also known as Hypervisor, can be used to enhance the security of VMs and applications running on those VMs. For instance, the VMM logs can be analyzed, not only to detect attacks but also to replay the execution of VMs to detect vulnerabilities of the deployed applications.

Although the new security architectures built on VMMs enhance the security and dependability of cloud applications, the VMM also offers a single point of failure. If adversaries can get control of the VMM, they can attack all tenant applications. VMMs themselves must be secured. For example, Amazon EC2 uses the concept of a security group where a particular rule can be applied to a VM to enhance security – for example, by restricting communication with the target VM to a predefined set of IP addresses. Another approach is to use fully encrypted applications; for instance, Data61’s EncryptedDB® technology enables execution of encrypted SQL queries over encrypted relational databases on the cloud.

**Termination**

Secure termination of cloud applications remains a challenging problem. The secure termination not only needs to deal with the secure shutdown of VMs in due time, but also the secure archiving and deletion of data. When the VMs allocated for a particular application are shut down, we need to ensure that VM images are secured and the termination is dealt with properly so that no leakage of data occurs through CPU and memory pages.

The US National Institute of Standards and Technology (NIST) has recognized the issue of secure deletion of data as a key computer security challenge, and developed guidelines for media sanitization. This issue is further complicated in cloud storage services, because the owner doesn’t have full control over the data as well as the storage media. Techniques such as policy-based file-assured deletion and Proof of Erasability (PoE) have been proposed to address this challenge.

We addressed secure data deletion using a Key Management Service (KMS) – see the “TrustStore” sidebar for details. The basic idea behind the scheme is to decouple the keys and encrypted data, where the keys are managed by a key management service independently. Secure deletion is achieved by deleting the keys, which makes the encrypted data irrecoverable.

**Data Storage**

Data security is important not only during the execution of cloud applications, but also during other phases of the SSLC. The data must be protected from both remote and local adversaries while in motion, as well as at rest. Data security is often defined as the CIA properties – that is, confidentiality, integrity, and availability.

Techniques proposed to achieve confidentiality of data fall into three general categories: architecture, privacy, and security. Hybrid-cloud techniques are in the first category, in which the sensitive processes and data are always kept in a private cloud and non-sensitive processes and data are sent to a public cloud. K-anonymization and differential privacy are examples of two key privacy-preserving techniques. The third category includes cryptographic techniques used to protect confidentiality of the data, such as homomorphic encryption.

Data integrity in the cloud is a well-researched property within data security. The techniques proposed in the literature can be grouped into two categories: public auditing and verification. In public auditing, techniques such as homomorphic tokens and distributed erasure-coded data have been used. Verification is generally studied under two categories: Provable Data Possession (PDP) and Proof of Retrievability (PoR). However, there are two key shortcomings with these approaches: efficiency and reliance on the storage service.

High availability of data in the cloud can be achieved by using multiple cloud service providers and replicating data among them. Examples of such systems include HAIL and DepSky, which uses the Cloud-of-Clouds principle. Our solution, TrustStore, addresses some of these challenges (see the sidebar for details).

In summary, the life cycle of the SaaS from development to termination – including the data consumed and generated – should be secured so that if things go wrong, we can detect and audit violations, and find the person who is accountable.

In terms of dependability, the cloud offers the opportunity to back up whole applications and their data. One solution, which originated within Data61 as the start-up company Yuru-ware, is now available as the product Unitrends Boomerang. It allows copying all VM images, virtual disks, and configurations from one cloud provider to another – for example, from VMware-based clouds to Amazon Web Services. In the case of a cloud
The POD framework targets the dependability of cloud application deployment specifically. An overview is given in Figure A, which splits the view into offline and online phases. At the core, it uses cloud metrics and logs from operations tools, like the open source tool Asgard from Netflix. Such tools manage the deployment of an application to a group of servers in the cloud — for example, through a rolling upgrade where old servers are iteratively terminated and replaced with servers running the application’s new version.

In the online phase, we use current logs and metrics in combination with the annotated process model created offline. There are two POD-Detection services for this purpose. First, the conformance checker tracks if the behavior and timing of the current execution is in line with the model. Second, assertion evaluation tracks the effects of the current execution on the metrics, and uses hand-coded additional assertions to check against the cloud API if process steps really had the desired effect. If any errors are detected, POD-Diagnosis and POD-Recovery are triggered directly. POD-Diagnosis uses conditional probabilities and diagnostic tests to find the root cause. POD-Recovery takes actions to bring the process back to a normal state — for example, by terminating a server that is “hanging” during boot-up and replacing it with a new server.

A number of papers have been published on POD, including exhaustive experiments to test its effectiveness and efficiency. As a starting point, please refer to previous work.1,2

References

During good runs of such operations processes, we collect logs and metrics. These are used in the offline training phase, where we generate a process model for the operation from the logs, through Process Mining — a set of techniques originally developed for business processes. Besides the purely behavioral view, we also build precise timing profiles of activities. Further, we annotate the model with information on how metrics change while the process is happening, such as the number of currently running servers decreasing by one whenever a server termination event has been logged.

Figure A. Overview of the process-oriented dependability (POD) framework. Here, the view is split into offline and online phases.

www.computer.org/computingedge
One of the systems that aims to provide a secure storage system on the cloud is Data61’s TrustStore. Figure B shows the TrustStore architecture. It provides a thin middleware that ensures all data transferred from a client application to cloud storage services are always encrypted. Because it supports multiple cloud storage service providers, it enables high availability through Redundant Array of Independent Nodes (RAIN), where each cloud service provider acts as a logical node. Data integrity is managed through an independent service, called Integrity Management Service (IMS).

The key management service (KMS) not only solves the problem of managing keys, but also enables establishing the root of data security to the user, sharing data to support collaboration, and secure deletion of data at termination by destroying the keys permanently. Once the keys in the KMS are destroyed, the data stored in the cloud storage service are irrecoverable. The access policies are attached to the keys, and support policy-based store, access, and deletion of data. In addition, the IMS protocol can be used to implement the proof of erasability (PoE) by using hash values stored by it.

A number of proof-of-concept applications were built using TrustStore, including TruXy (a trusted Galaxy for scientific workflows) and CloudDocs (which allows secure sharing of documents over the cloud).

References

Ingo Weber is a senior researcher and team leader in the Software Systems Research Group at Data61, CSIRO, as well as an adjunct senior lecturer at the University of New South Wales (UNSW), Australia. His
research interests include cloud computing, DevOps, business process management, and dependability. Weber has a PhD in computer science from the University of Karlsruhe (TH). Contact him at ingo.weber@data61.csiro.au.

Surya Nepal is a principal research scientist in Data61, CSIRO, Australia. His research interests include cloud computing, Big Data, and cybersecurity. Nepal has a PhD in computer science from Royal Melbourne Institute of Technology, Australia. Contact him at surya.nepal@data61.csiro.au.

Liming Zhu is the research director of the Software and Computational Systems Research Program at Data61, CSIRO, Australia. He also holds conjoint positions at UNSW and the University of Sydney. His research interests include software architecture, dependable systems, and data analytics infrastructure. Zhu has a PHD degree in software engineering from UNSW. Contact him at liming.zhu@data61.csiro.au.

CALL FOR ARTICLES

IT Professional seeks original submissions on technology solutions for the enterprise. Topics include

- emerging technologies,
- cloud computing,
- Web 2.0 and services,
- cybersecurity,
- mobile computing,
- green IT,
- RFID,
- social software,
- data management and mining,
- systems integration,
- communication networks,
- data center operations,
- IT asset management, and
- health information technology.

We welcome articles accompanied by Web-based demos. For more information, see our author guidelines at www.computer.org/itpro/author.htm.

WWW.COMPUTER.ORG/ITPRO
Frequently Unanswered Questions

Gerard J. Holzmann

WHEN YOU’RE TRYING to find that bug that makes your code do something that defies logic, the best thing you can do to get unstuck is to explain to a colleague why the bug can’t happen. Your colleague doesn’t have to understand your code or do more than just wait for you to say, “Ah, never mind, I figured it out.” I’ve been on both sides of these “discussions”: as the developer trying to figure something out and as the listener waiting for that aha moment.

The key here is that you force yourself to explain to another human being the things that appear obvious to you, as clearly and concisely as you can. To do so, you have to find the essence of a problem, the key insights necessary to understand it, and why the code you wrote was designed to work that way.

The magical “please help me debug this” conversation is very different from the discussion you’d have if you were teaching someone to use your application. In the latter case, you’d likely concentrate on just the part the user needs to know, and you wouldn’t spend much time on the internal details. You’d more likely focus on the “what” than on the “why” or “how” of your application.

This brings me to the question of how we generally write and document code. Just about every non-trivial tool or technique you want to use must be learned, and often you don’t have the benefit of talking directly to the designers about the “why” or “how.” You can only look at the manual that spells out all parts of the “what” in excruciating detail. That can be a problem, because if you don’t first understand the “why” or “how” of a new tool, you might have considerable difficulty finding the answer to the mundane problems you might run into when trying to come up to speed. It can take a long time to develop the intuition that makes all the puzzle pieces fall in place before you can answer the low-level questions.

Once I start writing the manual for a tool I’m developing, I often go back to the code to restructure it so that explaining how it works becomes easier. Sometimes we take short-cuts in an implementation, creating exceptions to more general usage principles that apply only in special circumstances. You as the designer will have no trouble remembering those quirks and why they hide in the code, but things change when you have to explain them to someone else. More often than not, it’s easier to go back to the code and remove the quirks so that nothing needs to be explained or remembered. Clearly, doing so improves the code and simplifies the documentation. The only reason it happens is that you make yourself explain things to a new user by writing the manual as clearly as possible. It’s this process of explaining that helps reveal the dusty corners and the less-than-perfect logic you used in writing the first version of your code. The most assured trigger you have to clean it up is to force yourself to explain it.

Thinking in Phases

We often think of software development as an ideally streamlined process consisting of three phases: design, build, and test. If we want our code to be useful to anyone, though, we’ll reluctantly have to acknowledge that we also need a documentation phase, which typically comes at the very end. Generally, there will be some iterations across these phases before we get everything right. Figure 1a illustrates the idealized process.

If writing clear documentation can affect code structure in the way
I am suggesting, this classic software development model perhaps isn't the best way to produce good code.

Larger companies can afford to employ armies of people who are good at producing volumes of user documentation. Sadly, many of these professional documenters don’t really understand the code they have to describe. Usually, this produces overwhelming amounts of poorly organized text that users must search to see if it happens to cover the questions they need answered. And, if I’m right, the lack of early documentation can also result in substandard code.

I was reminded of this phenomenon when I installed a new version of a popular static source code analyzer on my system. It ended up consuming an astounding 2 Gbytes of disk space, with close to 200 Mbytes of documentation. It’s a fair bet that no user will read through all the documentation before using the tool—until he or she gets stuck, of course, and needs a specific question answered. Then, the goose chase will begin, to find the answer in the wealth of text. If the user is lucky, a text search will find it. More likely, though, the specific instance of the problem the user ran into won’t be described anywhere and must be debugged with the help of someone who actually understands the code.

The Guessing Game

Not all applications come with megabytes of documentation. It’s either feast or famine. Sometimes, the only documentation will be an online FAQ, which is perhaps best described as a modern version of the guessing game. It’s similar to how a help line is typically organized. If you call a help line, you’ll likely end up in the clutches of an algorithm that tries to guess why you’re calling. “Press 1 if you’re calling about X; press 2 if ….” You end up listening to all the guesses before you can make up your mind about the one that came the closest to your problem. And then you’ll have to start all over again.

If your laptop fails to connect to the Internet, is that a hardware problem, a software problem, a network problem, or an authentication problem? Well, my guess is that you don’t know, and that’s probably why you’re calling the help desk. But before you can speak to a human, you’ll have to win the guessing game. If you guess wrong, you’ll just have to call again and try a different response.

Write the Manual First

I suspect that the reason for the poor quality of most tool documentation is that it’s written much too late in the software development cycle, and often by people who don’t know much about the actual code. Perhaps, if we started producing the documentation first, especially the user manuals, before we start writing code, the quality of both the documentation and software would improve.

In addition, the quantity of the documentation would shrink to something that can actually be useful to ordinary mortals. If the documentation is written early, the “what” details aren’t available yet, forcing the documenter to get the “why” and “how” right. The user will then more likely find a logical structure that provides the insight needed to resolve the conundrums that otherwise defy explanation.

Good documentation, like good writing, is typically short and to the point. Some of the most influential papers in science were remarkably short. For instance, Peter Higgs’s 1964 paper postulating the existence of what we now call the Higgs boson was barely two pages.1 In our own field, Gordon Moore’s 1965 paper describing what we now call Moore’s law was less than three pages,2 and Edsger Dijkstra’s famous “Go To Statement Considered
Harmful” from 1968 was just a little over one page.3
If you started programming early enough, you probably learned C from Brian Kernighan and Dennis Ritchie’s 1978 book The C Programming Language.4 The main text of the book’s first edition counted just 178 pages, plus an appendix with a full C reference manual of 40 pages. The book didn’t start with a list of keywords; it started with the famous “hello world” program to ease readers into the language. Today, books seem to fight for shelf space with ever-increasing page counts. For example, a recent C++ textbook has 1,368 pages and a spine measuring close to two inches. The 65-page index to that book is larger than the full C reference manual from Kernighan and Ritchie’s book. I know which type of book I’d rather learn from.

Test-Driven Development
Many have observed that the build and test phases shouldn’t be separated as cleanly as Figure 1a seems to claim. In practice, the two phases will always overlap. A good case can further be made that test development should start even before any code is written. That is, the initial part of the test phase would precede the build phase. This approach is called test-driven development, with the mantra “Write the test before you write the code.” When you develop code this way, initially all tests that have been defined will fail. But then slowly, one by one, as your code starts acquiring the functionality it’s meant to have, the tests will start passing. When all tests pass, you’re done.

In large software development projects, the design phase should result in a list of requirements the final code must satisfy. There’s no reason why you couldn’t develop the test suite at that point, which should eventually be able to show that each requirement was met. Documenting the requirements and tests up front will force you to be clear about the intended performance envelope of an application. That in turn can let you make an informed assessment of the risk that the application will find itself outside that performance envelope when conditions aren’t favorable.

Clear requirements and clear documentation let us build high-level models of code that clearly describe the underlying design. This has inspired the approach of model-driven design, which is quickly gaining strength in industry. The biggest gain model-driven design brings, though, isn’t the specification of a product itself but the ability to analyze it with powerful tools such as logic model checkers.

If we make documentation and modeling integral to design and make testing part of the build-and-code process, software development will start looking more like Figure 1b.

References

GERARD J. HOLZMANN works at the Jet Propulsion Laboratory on developing stronger methods for software analysis, code review, and testing. Contact him at gholzmann@acm.org.
Keep up with the latest IEEE Computer Society publications and activities wherever you are.

Follow us on Twitter, Facebook, Linked In, and YouTube.

@ComputerSociety, @ComputingNow
facebook.com/IEEEComputerSociety
facebook.com/ComputingNow
IEEE Computer Society, Computing Now
youtube.com/ieeecomputersociety
The proliferation of digital technologies and their increasingly common connection to the Web give rise to new opportunities for areas of study—in academia, business, and government alike. Web science and data science are two such examples, and although they are distinct, they share several similarities and can complement each other to gain greater insights into a huge range of topics. The Web presents such a variety of research challenges that an equally broad selection of techniques is required to gain insights. Since its emergence, Web science has been establishing a set of scientific methods for studying the Web's social and technological networks. This is enabled by the scale of data now available online. Social networking sites, webserver logs, and user-generated content are just a few examples of how the Web enables the creation of data at a rate that demands new ways of thinking, handling, and reporting findings. In order to exploit the vast amounts of data now accessible online, one emerging discipline that is crucial for Web science to synthesize with others is data science, which is developing approaches for extracting information from data so that it can be used in decision making and problem solving.

**Principles**

Web science studies the sociotechnical relationship between people and the Web, examining not only how the technology behind the Web facilitates the various applications now running atop it but also the role of the people using it, how their lives are changed by it, and how they play a vital role in shaping and maintaining the Web for future generations.\(^1\) The technical layer—referring to the technologies behind the Web—makes up only a part of Web science's focus. Equally important are two further layers: the social layer, which emphasizes the people and the content they create and share online and how the networks among them facilitate this, and the market forces and policy layer, which relates to the interconnected economic and political factors that shape the Web's evolution. As such, Web science is as much about the networks of people brought together by the technology as it is about the underlying technological network itself.

Web science's interdisciplinary aspect is fundamental to its aims of establishing the Web (not just its underlying technologies) as an object to study. The Web's impact on society, and vice versa, receives equal importance and focus. Sociology, politics, law, economics, and anthropology all provide invaluable contributions to the field and are fundamental in ensuring a holistic and societally beneficial analysis of the Web. Vast amounts of data produced by and stored on the Web reveal previously unobservable human phenomena at a grand scale, allowing new insight into society. This data fuels new research questions and possibilities spanning multiple disciplines and methodological approaches.

One approach to analyzing this data could be to draw on data science, which provides methods for dealing with data of all shapes and sizes and is proving useful for handling the rise of big data—large amounts of data that might be structured, semistructured, or unstructured. Big data is produced at such a rate that it requires new management policies and analytic methods, but it also offers the chance for new insights that capture phenomena in real or near-real time. The rate
at which data is created leads to challenges in storing, managing, and using the data productively. Much of this data is produced thanks to the increasing ubiquity of the Web, digital technologies, and connected devices, and it is the potential of unstructured data that can be highlighted as a primary difference between data science and established disciplines such as statistics.2 The demand for data science skills stems from research, commerce, and government alike. As with Web science, data science requires significant understanding of the subject area to complement programming and statistical skills to ensure a relevant question is asked and answered. New expertise is required, now not only limited to statistics but covering a range of topics from data curation to data analysis, data reporting, and storytelling.

Both of these disciplines have arisen through the prominence and increasing pervasiveness of the Web and digital technologies. They can provide new insights into a rapidly changing world and complement each other. To better portray the areas in which they overlap—along with those where they do not—this article will outline their similarities and differences.

**Interdisciplinary Approach**

Web science and data science are both interdisciplinary subject areas. For Web science, this aspect is key to the discipline and helps to produce an assessment of both the Web’s social and technical networks, going beyond what any one subject can produce. The disciplines involved can include syntheses of the humanities, social sciences, hard sciences, and engineering, and could include an assessment of the underlying technology and how this affects or is affected by a phenomenon occurring on top of it. Indeed, discussions have persisted about where Web science’s role lies in the computing discipline because its breadth goes far beyond the traditional focus of the subject.3

Data science focuses on a core statistical and analytical approach to data, which forms the basis of the discipline, and is assisted by techniques from computer science and its subdisciplines, such as machine learning and AI. These techniques are further complemented with domain insight into a problem, which helps contextualize and design the study, taking into account (where applicable) certain disciplinary aspects that provide direction for the statistical work. Data scientists will understand “the mathematics, statistics and physics necessary to integrate scientific algorithms into efficient architectures.”4 Following this, expertise is required to report on the findings in a way that makes sense to any invested parties, and this brings in practices from business and visualization; one of data science’s key roles is arguably to help understand business and innovation practices. As such, although both subjects share an interdisciplinary backbone, the nature of this mix is different: data science bases its approach on a key set of techniques focusing on and incorporating statistical data analysis, whereas Web science integrates a mix of approaches from across any discipline. Each way produces results that are grounded in the context of the problem they seek to solve, which is essential in considering the complex digital world that has arisen. Both disciplines therefore offer much in terms of the value of the insights that they produce. Web science is constantly aware of Web protocols and technologies, looking at how developments around these can impact human activity—and vice versa. The scale of the Web’s impact in the contemporary world means that this touches on most aspects of modern life. Data science can go further in some regards to produce insights into many industries or disciplines where the data involved has little to do with the Web itself—or the people involved in using and shaping it—and as such is more technology agnostic.5

**Actionable Insights**

Both data science and Web science help produce insights above and beyond what individual, traditional disciplines could provide; the key factor is that these insights are actionable and can be used to identify social or corporate phenomena, implement new policies, and make business decisions. Some existing subject areas will rightly question whether everything is completely problem-driven and actionable, but the affordances of studying data to this scale can still offer valuable opportunities for complementing existing research practices.

In Web science, studying interaction and communities on the Web facilitates new possibilities for understanding a population’s social and cultural norms, upon which new policies can be designed that consider both the Web’s technological affordances and the social practices these facilitate. The subject has an established goal associated with it: to ensure that the Web remains beneficial to society, meaning that insights are gained that contribute to this aim.6 In data science, the data might not be Web related and the goals
could be wider than this particular sociotechnical artifact, but studies will often utilize several Web-based technologies, such as semantically tagged datastores, in order to store and process that data. The insights can then be used to develop new business models, identify gaps in a market, or spot trends or anomalies in anything from agricultural data to security data to healthcare data. Similar to network science, which examines not only the Web but also transportation, biological, and other technological networks, data science produces insights that are actionable across a range of domains. It is clear, however, that many of the techniques and resulting insights gained from data science are applicable for use within Web science research. New policies could be implemented based on large-scale analysis of Web data that combines subject expertise and knowledge of social science with the computational tools of data science. In this example, data science’s insights and outputs are equally labeled insights from Web science; similarly, if the domain was astronomy, the data science insights would be astronomical outputs. Data science offers the tools and methods to produce insights, but it is within the context of the discipline—Web science, astronomy, or any other—that the insights are actionable.

Research Methodologies

Given the broad range of disciplines with which Web science seeks to integrate, it is unsurprising that there is a mixed set of research approaches. Mixed methods combining interpretive or constructivist approaches with positivism are common and emphasized as essential, and they help Web scientists understand aspects of the Web such as online human behavior, social network formation, and the spread of viral content. Alternatively, being more closely tied to certain disciplines, data science often has an empirical element at the heart of its methodology. However, new methods and approaches are required compared to traditional statistical approaches, because the data is more complex and is often being created, accessed, and analyzed in real time to provide instant analysis. Given this data’s relevance for assessing social phenomena, business insights, and policy provision, however, it is also necessary to contextualize these results, and accordingly, the research design must have an investigative and interpretive element. Therefore, mixed methods is again a suitable approach, but the way in which the methods are combined and the permanent inclusion of an empirical element demonstrates the differences with Web science.

Data science also draws heavily on open innovation, in which data is made available to external researchers or analysts and new ideas are sought. This increases the chance of novel approaches being implemented to produce insights, leading to a greater opportunity for flexibility in the methodological process.

Research Agendas

Web science has been described as being “focused on how we could do things better.” This refers to the potential to improve understanding about the Web and to ensure that future developments remain beneficial to human society and that the Web remains “pro-human.” In their “Manifesto for Web Science,” Susan Halford and colleagues propose that Web science should go further than simply being “a sociology or a computer science of the web,” suggesting that it must be a genuine intersection of disciplines that can examine micro- and macrophenomena on the Web itself. Web science research can therefore take numerous forms, seeking, for example, to gain new insights for political decision making and business practices. This could involve analyzing social media data on community and interactions; investigating user journeys, roles, and experiences with online services; and establishing guidelines around privacy, net neutrality, and security. For example, MIT’s Internet Policy Research Initiative (http://internetpolicy.mit.edu) emphasizes public policy research, highlighting the way in which the Web science approach differs to that of data science.

Data science can be based on anything from the social sciences to healthcare and focuses on problems that arise from large amounts of data and the computational overhead required to process that data.
assessing and planning Web development. However, data science’s agenda goes much further; for example, ChrisMattmann discusses these techniques’ use and importance in astronomy and Earth sciences, areas in which the amount of data collected can reach phenomenal levels, and new techniques are required in order to make progress.

Data science clearly offers much to other disciplines’ research agendas. Although Web science seeks to improve understanding about all of the Web’s technical, social, and policy-based layers, data science offers new opportunities to gain insights into existing research initiatives in any discipline in which data is becoming—or has already become—prevalent. It seeks to develop the approaches to analyze this data, ensuring that it can be used to gain as much value as possible. The resulting techniques typically can be applied to numerous scenarios, increasing the potential for further value to be exploited elsewhere. Data science contributes new methods for knowledge discovery from ever-growing amounts of data; the discipline’s overall agenda therefore relates to supporting other disciplines to ensure that their datasets can be analyzed, exploited, and used to ensure that knowledge can be extracted, regardless of the data’s size or complexity. As Web science does in relation to Web-related issues, data science seeks to ensure that policy decisions and business insights can be made across domains and that important scientific questions can be asked—and answered—from the growing stores of data that organizations of all sizes now have access to.

Given all this, data science clearly supports the ongoing development of Web science as a disciplinary lens through which to study the Web, and it helps to inform the design of new methodologies in the area—as network science does for techniques to study the underlying Web infrastructure and to understand social connections online. In order to answer the scientific questions posed about the Web, data science techniques are required to study the huge amounts of data available and extract the relevant knowledge. Owing to the rapid speed at which the Web is developing and changing, however, these techniques cannot stand still; therefore, the development of the data science agenda is driven by the changing needs of Web science, and other disciplines, which in turn are afforded new opportunities to ask deeper questions and gain more valuable insights.

Data Analysis
Data science tends to follow a general pipeline of data collection, processing, cleaning, analysis or modeling, and then visualizing and reporting. Within this, analysis typically occurs at two points. First, exploratory data analysis is carried out on the cleaned data to ensure that it is fit for purpose; statistics and observations are used to ensure that there are no duplicate entries or absurd outliers, for example, and calculations are made to describe the data’s shape to ensure it meets expectations. Once this exploratory data analysis has been completed, the data scientist designs a model or algorithm to fit the data, depending on the type of problem and the question he or she is trying to answer. For example, this could be to classify a training dataset and develop a predictive algorithm to automatically classify future cases. Techniques include data mining to identify patterns and extract key relationships from the data variables and machine learning to improve predictive algorithms.

These techniques, understandably, also apply for large amounts of Web science research, particularly those employing the use of Web-based big datasets. Web science can employ data mining to notice trends in website usage, social media conversation, or information propagation. Longitudinal analysis of past behavior and machine learning can help model and predict future growth of a particular service or site or the future behavior of certain individuals. However, Web science can also go down more qualitative routes to carry out different types of analyses, employing coding and thematic content analysis to examine what is being said online or using an ethnographic study to observe an online community’s behavior. Web science is by no means limited to using just data science analytic methods, but they certainly provide an opportunity to use and exploit large amounts of data that could reveal trends and phenomena that would otherwise go unnoticed.

Observing the Web with Data Science
We have highlighted several ways in which Web science and data science differ, and we have shown where the two disciplines overlap and intersect—that is, the areas in which data science
techniques and approaches produce the insights and results needed to complement other methods in a Web science context.

As part of their research, the Web Science Initiative is developing a worldwide network of Web science observatories: a "global-distributed resource with datasets and analytic tools related to Web science." The Web Science Trust website describes a Web observatory as "a system which gathers and links to data on the Web in order to answer questions about the Web, the users of the Web and the way that each affects the other" (see http://webscience.org/web-observatory/list-of-web-observatories). A Web observatory is, therefore, a global resource to provide both an archive of shared datasets and open analytics tools to facilitate Web science research.

As a simple definition, a Web observatory could be seen as a platform for carrying out data science on Web-based data, which often concern the Web itself. Tools can be offered to support the entire data science pipeline of collecting, curating, managing, analyzing, and reporting on the data. Additionally, many studies about the Web are now becoming interested in real-time data analysis in which the data pipeline becomes a responsive and dynamic lens on Web data, allowing Web scientists to observe and respond to emergent social phenomenon. Addressing these needs, real-time Web observatories are being developed to offer a technical framework to archive and analyze data simultaneously, in an extensible manner. For data science itself, several initiatives have aimed at producing similar resources; for example, IBM's ManyEyes, which ran from 2007 to 2015, allowed anyone to upload, visually analyze, and then share their data (www.computerworld.com/article/2930326/data-analytics/ibm-to-shutter-dataviz-pioneer-many-eyes.html). However, although Web observatories heavily emphasize reproducibility, data science offerings have focused on this less, so this is an area that can be exploited to improve the resources available to the discipline.

The Web observatory could offer an example of where data science can be used to gain insights into Web science research areas, as depicted by area 3 in Figure 1. Although the disciplines differ in scope, this intersection is an area in which the two could be used together to produce answers about the Web. As we mentioned earlier, a Web science study could focus on completely qualitative data, for which this approach would be less suitable, however, for many studies—especially those containing quantitative social network data, weblogs, and longitudinal records of activity—the use of data science and an understanding of how to manage the volume and velocity of data involved could provide a Web scientist with the tools with which to study the Web. In other cases, data science approaches would not be suitable for the heterogeneity of the datasets often investigated in Web science—whereas in data science the data all tends to come from a particular subject domain, the interdisciplinary mix in Web science means that the data is often about a mix of concepts. Frequently, this involves data that is about people and their activities online, introducing a plethora of issues around privacy, security, and trust; therefore, rigorous access controls, ethical policies, and terms of conditions must be designed and implemented. These factors together introduce barriers to sharing data, which subsequently require a decentralized and distributed approach to solve—when using data from multiple organizations, it is often prohibited to move this data between servers. As an additional barrier, Web science data tends to be dynamic—not just from the perspective of the rate of creation but also from the point of view of data management. Users can delete or remove data, which causes issues regarding the status of any collected or archived datasets; updates are required to keep them in line with these changes.

This discussion highlights some of the opportunities for data science within Web science observatories, but it also presents a range of important issues. These indicate areas where the two disciplines currently differ, but they also offer potential areas where the insights and best practices from Web science could be used to guide, influence, and inform the data science approach.

**The Intersection Between Web Science and Data Science**

Figure 2 displays the overlap of problem areas for which data science and Web science can be used together. The annual Web science conference offers a broad mix of topics, including behavior analysis, social network analysis, social science, and studies from the humanities, demonstrating that the scope of projects relating to the Web is itself massive. Data science could tackle further problems in which the data is not necessarily about the Web but uses Web technologies to store, manage, and distribute data in a way that ensures it can be used meaningfully.

---

*We have discussed the disciplines of Web science and data science as two distinct subject areas and have outlined*
the differences between them in terms of their approaches to and scope of study. However, there is a clear relationship that highlights areas in which these disciplines intersect and can complement each other significantly to achieve high-quality results in either field.

Data science can use Web science theory as a guiding influence on its approach to a particular problem. A key area in which data science goes beyond mere statistics is the domain knowledge that informs it and the actionable insights that are produced from the research design, which require an understanding of the context in which the results will be used. The use of a Web science approach to shape a data science study could reap rewards because the general strategy of mixing disciplinary perspectives to investigate the issue could provide the grounding and context to ensure that the data science itself both benefits the problem and produces results that can help alleviate the problem. For example, studying a large medical dataset about asthma symptoms requires expertise to be able to ask the correct question of the data, understand what the data is really showing, and ensure that the insights gained are accurate, validated, and verified by their existing domain knowledge.

A further opportunity for intersection and synthesis between these disciplines is for data science to be used as the quantitative element of a mixed-methods Web science study, in which the data in question is about the Web in some way, and the data’s volume or velocity makes it difficult to analyze without following the data science process. This approach might use resources such as a Web observatory, which offers a platform for storing this type of data for use in research about the Web, and demonstrates the overlap between a science that seeks to establish new and evolving ways to handle data and one that seeks to study the very sociotechnical phenomenon responsible for producing much of the data itself.

References

Christopher Phethean is a research fellow in the Web and Internet Science Group at the University of Southampton. Contact him at c.j.phethean@soton.ac.uk.

Elena Simperl is an associate professor in the Web and Internet Science Group at the University of Southampton. Contact her at e.simperl@soton.ac.uk.

Thanassis Tiropanis is an associate professor in the Web and Internet Science Group at the University of Southampton. Contact him at t.tiropanis@southampton.ac.uk.

Wendy Hall is a professor of computer science at the University of Southampton and the director of the Web Science Institute. Contact her at wh@ecs.soton.ac.uk.
Cyber-physical-human systems (CPHSs) consist of interconnected systems (computers, cyber-physical devices, and people) “talking” to each other across space and time, and allowing other systems, devices, and data streams to connect and disconnect.\(^1\)

The *NASA System Engineering Handbook* defines a system as the combination of elements, including personnel, processes, and procedures, that function together to produce the capability to meet a need.\(^2\) Thus, to effectively build CPHSs with people in the loop, it is important to be able to describe what a person can do, and when and where he or she can do it within the CPHS.

Though most of us think in terms of people using systems, many complex systems (such as the smart grid or smart cities) are actually a combination of computers, machines, and people all working together to achieve system goals.\(^1\) Sometimes, the role of people in these systems is passive. For instance, when social networks are used as sensors, people are not active participants; rather, their general observations are analyzed for information useful in a specific situation. In other cases, people actively participate. A person might be required to perform maintenance on a piece of equipment or respond to a medical emergency. In such cases, it is important to understand what capabilities the person who might perform the task has. Many have predicted that human-interactive systems, such as CPHSs, will continue to leverage people’s capabilities—in particular, high levels of situation awareness and adaptability—to better meet the goals of advanced complex systems. A big challenge is how to choose the right person for a given task.

**People in the CPHS Loop**

While most systems are designed to satisfy a person’s requirements as a user, people, in turn, can help a complex system make intelligent decisions and achieve its goals (which ultimately are people’s goals). Human-in-the-loop and human-in-the-mesh have extensively been used to describe the person as an operator and a part of CPHS.\(^3\) To develop a new generation of CPHSs that includes people in the loop, it’s important to understand what differentiates a person participating in a CPHS from a traditional cyber-physical component:

- **Cognition.** People have brains, eyes, and ears, whereas computer systems use digital processors, sensors, and actuators. The different ways in which people and computers observe, process, and act present challenges (and opportunities) for people to work together with computers to best achieve a goal.

- **Predicatibility.** People don’t perform the same task the same way every time; they might choose not to follow instructions or lose focus in the middle of a task. Although people might be less reliable than computers when it comes to following instructions, they have the ability to adapt much better to changing situations and can often come up with innovative solutions.

- **Motivation.** People, unlike computational systems, require incentives.
This motivation takes many forms, from monetary compensation to a pat on the back. Without proper motivation, a person might not perform a task even after agreeing to do so.

The challenge is that it’s difficult to know what a person is capable of doing—that is, what each individual can and cannot do in a given system, in a given context. Unlike mechanical sensor devices, people might choose not to do something, or choose to do it in a different way. For instance, an air quality sensor in a CPHS will always report the air pollution values in the same way, with the same uncertainties. On the other hand, a person observing might let their mood influence their perceptions of pollution when reporting, or might stop reporting altogether without notice. The systems designer must design the CPHS system with an understanding of what people can do and how they interact with other people, computers, and machines.

To help express this, a human service capability description (HSCD) data model is proposed that shows a person’s general capabilities and his or her ability to perform specific tasks (for instance, sensing, actuating, and processing). The capabilities are grouped into elements, so that designers can effectively identify and integrate people into their design process. The HSCD model uses the basics of service capability description, but is tailored to describing people.

**Human Service Capability Description Model**

In a typical CPHS, people will function together with other components, such as sensors, actuators, processors, data stores, and networks. The main objectives of the HSCD model, shown in Figure 1, are to represent

- the tasks a person can perform;
- the qualifications a person has for performing those tasks;
- the types of interfaces that can be used to interact with the person; and
- the identity of the person (from a system perspective).

In the HSCD model, a CPHS is a system that is composed of core components (including the human component) that are delivered as services. These components are sharable, configurable, and networked at every scale. They exist as service components that interact with the other CPHS components through digital communications. People can perform complex tasks based on their ability to sense, act, store, and process data.

The HSCD model describes a person’s capabilities using the following elements (italics are used to represent a class in the HSCD model shown in Figure 1): **Service, PersonDescriptor, InteractionEnd, Task, Availability, Qualification and Certification, Rating, and Authority.**

---

**Figure 1. Overview of the human service capability description model.** The model can be used to help understand a person’s capabilities and his or her ability to perform a task.
Service

Service describes the information required for a person to interact with other services. Each service has an identifier (uid), which can be referenced when the service is being requested by another component or application; a person who is represented by the service; and a set of interaction ends that describe how other services would communicate with the person.

PersonDescriptor

PersonDescriptor is the person who can perform a service. This person uses one or more applications to provide a connection between him or herself and other services (through some digital device). The Person is a collection of information on a person's qualifications and past experience in performing a given task.

InteractionEnd

The InteractionEnd is one end of the communication between two service components, which the person accesses through an application. It contains information related to the Data that can be communicated, as well as the way it can be communicated through its Interfaces.

Interface describes how the InteractionEnd used by the person exchanges information with other components, such as protocols, devices (such as mobile), and mechanisms used to communicate. It's important to understand how information can flow in either direction.

Data describes the format of the data being exchanged between the Person and other CPHS components—it covers data structure, format, and semantics. This is important because humans use different languages to communicate compared to computers.

Task

A Task relates directly to a task the person can perform in the CPHS. The task can be specific or general depending on the situation. A Person might be able to perform more than one task.

PerformedTasks are those a person or device has previously performed in the context of a CPHS. These are particularly important because it can be difficult to determine a confidence level for task performance. Because people can be rated based on the performance of previous tasks, this is one of the better ways to gain confidence in their ability to perform. Under a Rating, the performed task can be rated by the system builder or other components based on how well the person or device performed the task. Given that this reflects real-world feedback on the person's performance, it might provide the most relevant information about a person's ability to perform a given task.

PotentialTasks are those that a human is willing to perform in the context of the CPHS.

Availability

The person must be available to perform and complete a task—that is, the person must be in the right place at the right time to perform a task. The location is the place in which a person is available to perform tasks. It might be specific to a PotentialTask, or can apply to all the tasks the person is willing to perform. The time type describes the period in which a person is available to perform tasks. It might also be specific to a PotentialTask, or can apply to all the tasks the person is willing to perform.

Qualification and Certification

A qualification shows that the person has some general or specific aptitude. Qualifications are more formal than ratings and usually cover a set of tasks. The qualification types are as follows:

- Training is designed to contain any information related to the formal education or training a person has that is relevant to the tasks the person can perform.
- Testing contains any aptitude test results relevant to the set of tasks the person may perform.
- Authorization contains any authorization (including security) status related to the set of tasks the person may perform.
- Experience contains any job-related information, including recommendations from previous or current employers, commendations, and other relevant information.

A Certificate, therefore, represents proof of a Qualification.

Rating

A Rating is informal feedback given on how a person performed a given Task. Because of its informal and likely unverified nature, a single rating might not be useful, but a set of ratings should give the system builder some confidence in the person's ability to perform a Task. In the case of participatory sensing, individuals might be rated by their peers (other services), and the results could be used to gauge their ability to perform a task. Combining the Ratings with the Qualifications will hopefully provide a high level of confidence.

IssuingEntity

In the context of a CPHS, an IssuingEntity provides some level of assurance of the rating or certification of the person's qualifications. This can be an accreditation or standardization body, or an informal entity. The assurance level required will depend on the task at hand.

Evaluation of the Model

This model focuses on the structure of data representing a person's capability. Further input is needed from the community of experts
working on modeling human behavior. One next step is to work with experts in the fields of human behavior, human resources, and others involved in understanding how people act. Additionally, a testbed is being developed at the US National Institute of Standards and Technology (NIST) by which this model could be implemented. A possible test scenario for the testbed is the emulation of a disaster-response system. Based on a real-world scenario, a simplified virtual environment will be created with experimenters representing residents and emergency responders in a flood-affected region involved in disaster and recovery operations. The CPHS model uses physical and social information to direct emergency and rescue teams to respond to specific emergencies in the most efficient manner possible. Volunteers on the scene are then directed to help out where needed.

Research and development in ubiquitous and social computing—especially in the fields of the Internet of Things and participatory sensing and crowdsourcing for disaster management—is driving the need to be more explicit about how people are an integral part of systems and need to be included in the system-development process. The HSCD model is a good starting point for describing the roles people play in CPHSs and should help improve the CPHS design process. 

References

Sulayman K. Sowe is a senior researcher in the Information Services Platform Laboratory at the National Institute of Information and Communications Technology (NICT), Japan. His research interests include big data, cloud computing, data-intensive knowledge discovery, open source software development, knowledge sharing, information systems evaluation, social and collaborative networks, and software engineering education. Sowe has numerous publications in international journals and proceedings, and serves on a number of academic, review, and international committees. He received a PhD in informatics from Aristotle University, Greece. Contact him at sowe@nict.go.jp.

Eric Simmon is a senior scientist in the Cyber Infrastructure Group at the US National Institute of Standards and Technology. He leads the NIST effort in cyber-physical cloud computing and is cochair for the NIST Cyber-Physical Systems Public Working Group use case subgroup. Simmon is a member of the NIST Cloud Computing program, has worked with the international electronics manufacturing community to develop an international standard for exchanging material content data, and was an original member of the NIST Smart Grid Program (leading the architecture and vehicle-to-grid activities). Contact him at eric.simmon@nist.gov.

Koji Zettsu is a director in the Information Services Platform Laboratory at the National Institute of Information and Communications Technology (NICT), Japan. His research interests include information retrieval, databases, and software engineering. Zettsu received a PhD in informatics from Kyoto University. He is a member of the Information Processing Society of Japan, the Institute of Electronics, Information and Communication Engineers, the Database Society of Japan, and ACM. Contact him at zettsu@nict.go.jp.

Frederic de Vaulx is the vice president of Prometheus Computing and a senior software engineer, specializing in the analysis, design, and development of software applications, contracted to the US National Institute of Standards and Technology. He is involved in the NIST Cloud Computing Program, leads the Cloud Metrics Public Working Group, and has been involved in cyber-physical system projects at NIST for the past five years. De Vaulx received an MS in computer science from Ecole Supérieure d’Informatique et Applications de Lorraine (ESIAL), France. Contact him at f.devaulx@prometheuscomputing.com.

Irena Bojanova is a computer scientist at the US National Institute of Standards and Technology. She is the founding chair of the IEEE Computer Society Cloud Computing Special Technical Community, a cochair of the IEEE Reliability Society Internet of Things Technical Committee, and a founding member of the IEEE Technical Subcommittee on Big Data. Bojanova is the acting editor in chief of IEEE Transactions on Cloud Computing, associate EIC of IT Professional, and an associate editor of the International Journal of Big Data Intelligence. You can read her IoT blog and her cloud computing blog at www.computer.org. Contact her at irena.bojanova@nist.gov.

This article originally appeared in IT Professional, vol. 18, no. 1, 2016.
What Does a Skunk Works Do?

SHANE GREENSTEIN
Harvard Business School

When an established firm aspires to experiment in a radical direction, management gurus recommend opening a skunk works—an organizational home for high-priority original thinking and projects. It is housed away from the organization’s main operations, sometimes in secret or with organizational barriers. Typically the projects involve something of value to the future but are not directly connected to the present operational or service missions. Sometimes a skunk works has the approval of senior management, and sometimes it does not.

The phrase “skunk works” originated from the aeronautics industry, and in that context it had a specific meaning (and still does). The meaning of the phrase has evolved, and today it means something broader outside of aeronautics; that causes confusion, which further fosters poor managerial decisions. Today’s column maps the scope of change.

Appreciate how far the phrase has come. The very first skunk works began many decades ago with a project for the air force at a division of Lockheed in Burbank, California. It initially had called itself the “Skonk Works” in a bit of salty humor about its own secrecy. The phrase came from Al Capp’s Li’l Abner cartoon—the skonk works was a “secret laboratory” that operated a backwoods still. Eventually the label became well known. Too well known, as it turned out. Even in those days, publishers tried to protect their copyrights, and Li’l Abner’s publisher eventually asked Lockheed to stop calling itself that. Long story short, smell played a role thereafter, because the Lockheed facility was next to another plant that emitted fumes—like a skunk.

In the modern usage, a skunk works still seeks to depart from existing operational processes, but pursue a variety of goals. Despite that variety, all skunk works face similar issues. Without daily operations to ground routine, there are few measurable benchmarks that refer to existing operational metrics. That makes them challenging to manage and monitor, which makes it difficult to align costs with benefits. That is a fundamental economic challenge.

Science Model

One type of skunk works focuses on science, or, more commonly, its application to frontier engineering. Most computer scientists and engineers observe this model in university research or government laboratories. There the skunk works supports the discovery of scientific breakthroughs or facilitates elements of the scientific process, such as refining scientific implications, developing prototypes, and implementing initial frontier proposals at scale.

Measurement of progress takes familiar scientific forms for advanced engineering, such as papers published or patents applied for and granted. In short, there is a way to measure major incremental discoveries, and at intermediate stages of progress. This works well when these standard measures actually correspond with meaningful progress, which sometimes happens.

These types of skunk works tend to become institutionalized inside labs or, from time to time, small satellite organizations housed away from headquarters. For example, many large firms have started a skunk works in Santa Clara, sometimes to connect with researchers in the area. That follows a long history going back to Xerox PARC, which stood for Palo Alto Research Center. PARC hired some of the best computer scientists in the area at the time, and they invented some of the most canonical prototypes of the 1970s, including the mouse, LAN, and GUIs. (And, infamously, Xerox HQ had difficulty commercializing these inventions.)

There are many variants on this model. For example, some skunk works devote themselves to producing a “white paper” with employees who get temporarily relieved of regular duties. The white paper investigates a previously unexamined topic, and sometimes develops a prototype or a strategy for scaling the frontier. Although these projects might sound like “skunk-works-lite,” the organizational form serves its purpose, not letting organizational politics get in the way of the analysis.

The most famous of this “lite” model was the small group inside Microsoft
who investigated the Internet in the summer of 1994, before Bill Gates took much interest. The people in the works put their jobs at risk, and, luckily for them, eventually got the boss’s attention.

Research Community Model

Another model of a skunk works focuses on building research communities. Leaders focus on developing a research community with a broad mission. Individual projects, or their success and failure, matter less than the development of a group with an identity built around shared interest. Leaders develop communication channels for the community and initiate institutional support to make the community self-sustaining.

DARPA has used this model for many decades, and it can also be found at many other government agencies with a mission to develop external research. Although DARPA’s development of the Internet is the best-known example, there are also variants outside of government, especially in engineering fields where evolving standards require frequent updates and revision. That is why, for example, Vic Hayes, who led the first committees to develop what became Wi-Fi, focused on developing his community into something self-sustaining. That helped the standard persist and grow.

Related indicators of progress include size of membership and other milestones related to its sustainability. Indirect benchmarks include the members’ prestige, symptoms of their devotion to their activity, and the spread of mission to groups outside of the core initiators.

Because some community members contribute without expecting direct compensation, assigning responsibility and credit can become an issue. For example, think of the Mosaic browser team at the University of Illinois in the mid-1990s. This small team of programmers built on top of the rapidly diffusing web and many years of accumulated processes for the entire infrastructure, and the team generated visible gains and received lots of publicity. Who should have gotten credit for this advance—the software team who built Mosaic, the research institute that paid their salary, or all the others who made the Internet reliable in the prior decades? Two decades later, many of those involved still argue about these matters.

Lead User Model

In a third model of a skunk works, the works acts as a lead user. Governments use this model to meet a unique government mission, such as a military need or NASA’s early efforts. The lead user pays for pushing the scientific and engineering frontier.

Lead users tend to measure progress with new metrics developed during experience. Certainly this was a feature of the situation during the NSF sponsorship of the Internet, when it was acting as a lead user. The management had visible symptoms of growth to show others, but it was challenging for them to explain the value of what they were developing.

More generally, lead user efforts are benchmarked against aspirations, and any benchmark compromises between the practical and the dream. Nothing challenges dreamers more than learning from (bitter) experience that their goals need to pivot. Leaders have to be visionaries and good managers. Those people are rare.

Many startups act as glorified versions of this type of skunk works outside of government, often with the explicit goal of being acquired later. Why did these lead-user projects get housed in a startup and not as a skunk works in an established firm? Usually there are dozens of reasons.

Kickstarter is full of firms like this. A recent known example is Oculus Rift, which pushed the envelope just enough and hit it big with an acquisition. Occasionally, a major venture capitalist will take on one of these projects. An example is SpaceX, which decided to develop frontier engineering and processes for reusable rocket parts.

Moon Shot Model

There is a fourth model of a skunk works. The organization defines an audacious target—it shoots for the moon. The audacious target acts as a unifying goal for a team and every participant. You will notice that this model contains elements found in all the others. Leaders develop communities, the participants tend to develop frontier engineering using science, and the organization also acts as a lead user.

Private firms occasionally like to shoot for the moon. Google used this type of organization to start its autonomous car project, and IBM used one to beat Jeopardy! with Watson.

The best part of this model arises at the end. It is easy to measure when it is done—a goal is either achieved or not. It can be extremely challenging, however, to indicate progress until that moment. For example, the team that developed Watson started as a scientific skunk works, as a part of IBM Labs. Because progress had slowed, management decided to convert the skunk works to the moon shot model. Hindsight makes this conversion look smart, but it was not easy to put all of the team’s eggs in one basket. It took the team four years to meet its goals. In the meantime, they published nothing, and they almost failed their trial with the Jeopardy! producers. Kudos to the team leaders for keeping this together.

Even when skunk works successfully meet their targets, they face issues transferring their success into regular operations. Firms and government both tend to put off the hard questions affiliated with such integration. That is a difficult topic, and one best left for another day.

Suffice it to say, there is no magic to integration. Because the projects deliberately varied from existing processes, making it work with everything is usually an effort as large as the skunk works itself.

Shane Greenstein is a professor at the Harvard Business School. Contact him at sgreenstein@hbs.edu.
The Amiga story began via a startup company with a vision, Hi-Toro (which later changed its name to the Amiga Corporation). The technology that sprang from Amiga eventually got absorbed into Commodore—a company at war with Atari and itself. As an internationally diverse group of users gravitated toward Amiga computers, problems in Commodore also escalated, eventually to a point of no return.

The Amiga was an American product that was hugely successful in Europe. It was the first multimedia computer that was also affordable. In the United States, however, the majority of users were in the entertainment industry because the Amiga was the only affordable computer that had graphics capabilities and video timing built into it. The turning point for this user trend was the Video Toaster, which was eventually released by NewTek for the Amiga 2000 in December 1990. That instantly made Amiga computers the go to system for high quality, yet affordable media editing and special effects. At the same time, the Amiga 500 had acquired a reputation for being a gaming computer in Europe. It was a serious rival to video-game consoles released by companies like Nintendo and Sega.

The community that built up around the Amiga was strong—so strong that when Commodore imploded the development base for Amiga computing almost immediately shifted from the US to Europe. Operating systems like AmigaOS 4 continue to come out of Belgium, and even today new hardware is being developed in places like Germany, Italy, and Great Britain.

Early online communication tools such as AmiRC were also put in place to support worldwide chatter. In fact, chat rooms, bulletin boards, and online forums have created the foundation for developer networks. Otherwise disconnected enthusiasts, who are detached from one another on a global scale, can come together to form a network online. They have kept the Amiga dream alive.

What can be considered old technology has been kept alive by users, enthusiasts who continue to find new ways to extend the life of otherwise aged computer chipsets. For example, with the aid of PowerPC accelerator boards, the last Amiga computers produced by Commodore are still being expanded in order to extend their lifespan. The original Amiga system architectures were flexible enough to enable users to hack, modify, and upgrade the system as desired. Despite a lack of advertising and in the absence of manufacturer-driven guidance, communities of users who still proudly call themselves Amigans have built up around these personal computers from the 1980s and 1990s. Community members have become developers, communicators, and user creators for older or niche solutions. New developments for these old computers continue to emerge to this day.

**Motivations for Continuing Development**

Prototyping and testing next-generation Amiga computers (such as the AmigaOne) can be expensive when conducted on a small scale. However, this development initiative and effort typically is not done in hopes of a return on investment, rather it comes from people “with a mission.” The motivation is probably not simply nostalgia for keeping the old Amiga machines going—plenty of machines have become obsolete and forgotten by their one-time users. Partly the motivation may be to keep the Amiga community and style of computing alive, a type of creative computing that is geared toward artists, videographers, graphic designers, and the like in addition to the technically minded.

More than that, the motivation seems to include remembering and revisiting ideas otherwise buried in “modern” computing. For example, there are people emulating Amiga chipsets in freely available software like WinUAE as well as community preservation packages like Amiga Forever from the Cloanto Corporation. (These can be used on current computing platforms without having to buy new hardware.) One might say that retrospective or retro computing is about progress and is driven by recognizing, understanding, and building upon past achievements.

Within the user communities of retrocomputing platforms like Amiga (networks that are active on a global scale) are dedicated individual champions of the community’s mission. One such individual is Trevor Dickinson, a geologist by training, business angel in his day-to-day life in New Zealand, and cofounder of A-EON Technology, a company that manufactures AmigaOne systems. Despite the fact that development money from corporate America dried up long ago, Amigans like Trevor have kept what might be considered an otherwise defunct platform going. They are developing new hardware, building Amiga-based and Amiga-like operating systems for PowerPC and x86 PCs as well as fully utilizing social media for worldwide brand communications.

**Other Aspects of Amiga Retrocomputing**

In addition to developing updated operating systems and hardware, Amiga users also breathe new life into
aging chipsets. For instance, PowerPC-based expansion hardware was first developed by Phase 5 Digital Products after Commodore had gone bankrupt in 1994. Developers are still driven by a philosophy introduced into their community by the original Amiga development team. Their line of thinking is to generate computing solutions that are consistent, simple, and enjoyable by design. When the user interface named Intuition for the original Amiga 1000 was developed, interactions between the user and the computer achieved this goal. As Director of Intuition, RJ Mical still believed this when talking about the system on its 30th birthday in 2015: “In my opinion, the Amiga gets to claim its place in the development of technology in modern civilisation.”

In this instance, the lessons learned have become apparent to retrocomputing enthusiasts over a notable period of time. Computer hardware, video-game systems, and operating systems are being kept alive by a core group of users. Progress is no longer defined by a traditional company or corporation-driven business model. Instead, ongoing developments are an unforeseen byproduct of a consumer culture, which has evolved beyond the intended commercial life cycle for a product.

In the place of centralized development workflows are enthusiastic user-creators located all over the world. These people have remained loyal to a technology when mass appeal has shifted to the next-best thing. As a cofounder of the company that developed the AmigaOne X1000 PowerPC, Trevor Dickinson is again a prime example.

The Amiga brand changed hands several times after Commodore went bankrupt. A nostalgia based spinoff company called CommodoreUSA even formed in 2010. It sold repackaged Intel PCs running a Linux-based operating system up until its founder, Barry Altman, died unexpectedly in 2012. Almost 20 years after the demise of Commodore, it was clear Amigans remained hungry for new hardware and software. Still lacking, however, was a sustainable development model that could make this a reality. It wasn’t until Trevor Dickinson formed A-EON Technology in 2009 that this came about. A-EON Technology even developed working agreements with other AmigaOne hardware developers, such as ACube Systems. In other words, sustainable development came about from within the Amiga community itself. Trevor wanted to support new hardware. As a result, a British company called Varisys soon designed motherboards called Nemo, in homage to Jules Verne, to run AmigaOS 4.

A-EON Technology released its first PowerPC-based computer, the AmigaOne X1000, in 2012. It was developed around a P.A. Semi (originally Palo Alto Semiconductor) dual-core CPU and Nemo motherboard. Because of A-EON Technology’s relationship with Varisys, XCore chips were also used for Nemo. The origins of this software-designed silicon technology—programmable semiconductors geared toward consumer-level manufacture—stemmed from a student project at the University of Bristol in England.

XMOS designed chips are one of many features used to make motherboards like Nemo easy to expand and customize. Expansion was first incorporated into “classic” hardware by people like Jay Miner, the engineer considered to be the father of the Amiga. In fact, it was this ability to expand and customize the original chipsets that enabled the long-term survival of Amiga-based computing in its wilderness years. The XMOS XS1-L2 124 chip was renamed Xena for the X1000 because Dickinson wanted to keep with tradition. He had been a devotee of Commodore-branded computers since the days of the 4032 (PET), and Amiga chips had been referred to by female names since Agnus, Denise, and Paula. This was the chipset designed for the original Amiga 1000.

Material quirks from the Commodore years were also added to the motherboard. For example, hardware engineers like George R. Robbins engraved B-52’s song titles onto their work. The Amiga 500 and 500+ featured “Rock Lobster,” the Amiga 600 had “June Bug,” the Amiga 1200 contained “Channel Z,” and the Amiga CD32 was “Spellbound.”

Trevor’s motivations for developing updated Amiga systems were personal and linked
to his deep understanding of the Amiga community. His aim to create new hardware has been further cemented through the release of the AmigaOne A1222, X5000/20, and X5000/40 in 2016.30 Developments were not primarily based on monetary gain, or his experience as a successful businessman: “To be perfectly honest, I really had no idea if the AmigaOne X1000 would sell or not. As a long time Amiga enthusiast, I knew there was a pent-up demand for more powerful modern AmigaOS hardware. After all, I wanted one myself!”31

Personal desire did not cloud Dickinson’s judgment, however. By the time A-EON Technology was formed, his experience in business steered design processes in the right direction: We took a phased approach to developing the Nemo motherboard for the A1-X1000. The first step was a proof-of-concept and we prefunded US $115K for initial development and prototyping. Five prototypes were manufactured for stress testing and porting AmigaOS 4. After review we took the decision to make a number of minor design improvements and create a revision 2 motherboard. Revisions cost more time and money and delay. Prototypes are also very expensive because they are made in very small volumes and can cost up to USD $5000 per board.19

The Cyrus and Tabor motherboards followed Nemo. They were used for the AmigaOne X5000/20, X5000/40, and A1222. Like Nemo, they were also named after characters from 20,000 Thousand Leagues Under the Sea by Jules Verne. Part of the development process for Cyrus included 50 beta testers from the Amiga user community. Incorporating XMOS technology into its design brings a flexibility of use familiar to anyone who has worked with programmable chips.32 The idea of using programmable chips to build new hardware was taken to a new level within the Amiga community by Dutch electrical engineer Dennis Van Weeran in 2006.33 He used a field-programmable gate array (FPGA) chip to create the MiniMig, which is open-source hardware that replicated the Amiga 500.34

A-EON Technology drew inspiration from all parts of Amiga history when designing its computers. For example, the AmigaOne X1000 combined expandability, implemented by the original Amiga development team, with ideas taken from independent developers like Van Weeran. In a more playful gesture, the B-52’s song titles “Breezin” and “Topaz” also made it onto the Cyrus and Tabor boards. Their predecessor, the Nemo motherboard, encouraged AmigaOne X1000 owners to “Keep This Party Going.” The Cyrus and Tabor motherboards also incorporated Freescale chips, much like the Efika series of motherboards produced by Genesi. (The move to Freescale by A-EON Technology was after P.A. Semi was acquired by Apple. In 2015, Apple announced that it may consider using its own custom chipsets for iMac and Macbook from 2016 onward.)

**Integrated Retail**

I see the timing of the resurgence in Amiga-infused hardware as feeding into broader changes taking place in the world of product development, marketing, and distribution. Information and communication technologies have increased the likelihood of independent or small-scale production taking place. For example, digital distribution sites like Steam provide an outlet for game developers.35 They enable user creators to have direct control over how they reach a customer and sell their products. This distribution model was also introduced into the Amiga community at Amiwest 2014, where the creation of Amistore was announced.36 Through the social networking aspect of such online stores, a shift in consumer culture has occurred that has been referred to as integrated retail by Sears-run iRLabs.37

**When Amiga Turned 30**

The first Amiga computer turned 30 years old in 2015. There were celebrations all over...
the world to mark the occasion throughout that year, including a two-day event held at the Computer History Museum in Mountain View, California, on 25–26 July 2015 (see Figures 1 and 2). As someone who attended Amiga 30 USA, it was interesting to observe the new information that came out of the event. Dave Morse, the businessman behind Hi-Toro and Amiga Corp., was seen as being a main driver for making the original Amiga development team what it was.38 Joe Decuir and Ron Nicholson, the first two hardware designers to work closely with Jay Miner, also shed new light on how the Amiga system architecture came about.39

New operating systems descended from the original AmigaOS were also on display at the 30th anniversary event at the Computer History Museum. AmigaOS 4, AROS, and MorphOS evolved from the last operating system released by Commodore, Workbench (part of AmigaOS) 3.1. AROS is open source and can run on most computing platforms. AmigaOS 4 and MorphOS are primarily PowerPC based.

Prior to the Amiga 30 events, there were also efforts to bring the user community together online. For example, a location-aware approach was developed in 2011. User maps created in Google Earth40 show where Amigans are based to encourage peer-to-peer engagement. Graphical representations of user numbers have also been created for different countries. These graphs show that the independent developer model now in place has resulted in a shift such that the majority of hardware and software development (and user culture as well) is now happening in the European Union, not in the US. That said, Googlers like Christian Stefansen have ported the Amiga 500 version of AmigaDOS to the Chrome Internet browser.41

**Conclusion**

Amiga technology came into being decades ago. The concepts (and some of the actual system components) had staying power despite the failure of the original manufacturer. Developers and users around the world continue to push and use Amiga derived and expanded technology. This is more than retrocomputing as a desire to simply reclaim past computing technology. This is an example of continued technology development by a community of invested users, a sort of consumer-led development.

**Acknowledgments**

This article is dedicated to Dave Needle, who passed away in February 2016. I also thank Glenn Keller, Dale Luck, Caryn Mical, RJ Mical, Mitchie, Joe Pillow, Stan Shepard, the rest of the original Amiga development team, Dave Haynie, Beth Arnold and the Commodore Amiga development team, Andreas Magerl and Amiga Future Magazine, Trevor Dickinson and A-EON Technology, the MorphOS team, the AROS team, Zach Weddington and the Viva Amiga documentary team, Aaron Ruscetta, Ravi Abbott, Andy Broad, Chris Collins, Stephen Jones, and Dan Wood.

**References and Notes**

4. The rights to AmigaOS 4 were granted to Belgium-based Hyperion Entertainment after a legal battle with Amiga Inc. ended in 2009; see [www.hyperion-entertainment.biz/](http://www.hyperion-entertainment.biz/).
5. Genesi (http://genesi.company/) created both the Pegasos and EFIKA PPC hardware to run AmigaOS and MorpOS.
8. AmiIRC (www.amirc.org) is the Internet relay chat client for all Amiga and Amiga-like computers.
9. The Aminet Archive (http://aminet.net/) is the largest and oldest archive for Amiga-related software and files online.
10. The Amiga.org website was originally a BBS started in 1988 by the North Alabama Society of Amiga Users (NASAU).
11. MorphOS (www.morphos-team.net) is lightweight Amiga-like operating system released for PowerPC hardware.
12. AROS Research Operating System (http://aros.sourceforge.net) was originally called the Amiga Research Operating System.
13. The Amiga Facebook page had over 150,000 likes by 2015; see https://www.facebook.com/amigafans.
14. PPC accelerator boards for Amiga computers were first discussed in the ESCOM period of ownership; see www.cucug.org/amiga/aminews/1995/at951111.html.
23. AmigaOS4 (www.amigaos.net) is the official website of the latest Amiga operating system.
26. XCore chips were designed for affordable manufacture via their programmable architecture; see www.xmos.com/products/silicon/l-series.
28. “Interview with Trevor Dickinson,” Retro Planet, 26 Aug. 2014; www.retroplanet.gr/content/interview-trevor-dickinson-%CF%83%CF%85%CE%BD%CE%AD%CE%BD%CF%84%CE%85%CE%BE%CE%87-%CE%BC%CE%85-%CF%84%CE%8F%CE%BD-trevor-dickinson.
32. XCore chips were designed for affordable manufacture via their programmable architecture; see www.xmos.com.
34. Open-source specifications for the Minimig reimplementation of the Amiga 500 are available at https://code.google.com/p/minimig/.
35. Internet-based distribution sites like Steam (http://store.steampowered.com/) have given independent game developers the ability to sell their products themselves.

36. A-EON Technology developed the Amiga orientated online store, Amistore (www.amistore.net), to help develop and sustain new software.

37. Integrated Retail Labs (iRLabs) was developed by Sears to evolve the relationship between store front and online shopping experiences; see www.searslabs.com.


40. Amigamaps (http://amigamap.com/) are spatially aware databases that were created by Amigans in order to help grow their user community.

41. In Dec. 2013, C. Stefansen used Google Portable Native Client (PNaCl) Technology to create a Chrome-specific version of the open-source universal Amiga emulator; see http://pnacl-amiga-emulator.appspot.com/

Adam P. Spring is a visiting lecturer in digital humanities and sciences at the University of Plymouth, UK. He was a postdoctoral associate at Duke University when this article was published. His interest in the Amiga community stemmed from his formative years in the 1980s to 1990s computer scene in the UK as well as his professional experience with high-end 3D visualization technologies and user communities. Contact him at adamspring@gmail.com

For this issue of ComputingEdge, we asked Rafael Prikladnicki about career opportunities in the popular field of software engineering. Prikladnicki, whose main research areas are distributed software development and agile methodologies, is director of the Science and Technology Park at Brazil’s Pontifícia Universidade Católica do Rio Grande do Sul. He is also the coordinator of the university's distributed-software research group and editor of IEEE Software's Voice of Evidence department.

ComputingEdge: What careers in software engineering will see the most growth in the next several years?

Prikladnicki: I would say that every software-related field will grow in the next several years, especially those involving medicine, smart cities, and data mining. For example, medicine is investing in a lot of technology to support healthcare. So there are many areas for which software will be critical. And this will increase the software engineering professional’s importance.

ComputingEdge: What advice would you give college students to give them an advantage over the competition?

Prikladnicki: I would say they have to combine software engineering theory and practice. Theory gives them what they need to start. Practice gives them the experience to stand above the competition. They can acquire this experience in several ways including, for example, partnering with companies that have summer internship programs.

ComputingEdge: What advice would you give people changing careers midstream?

Prikladnicki: The most important thing in my opinion is being satisfied with what you do, no matter what it is. If you are not happy with what you do, you are not performing at your best.
**ComputingEdge**: What do you consider to be the best strategies for professional networking?

**Prikladnicki**: There are several ways: participating in events (online or offline), planning who to contact and why, defining goals for specific contacts, exchanging business cards, and following up after a face-to-face conversation.

**ComputingEdge**: What should applicants keep in mind when applying for software engineering jobs?

**Prikladnicki**: While what they did in the past is important, what they want to do in the future is more important. They should avoid concentrating too much on their own history and focus on expressing their views and goals for the future and how companies can benefit from their knowledge and experience.

*ComputingEdge*’s Lori Cameron interviewed Prikladnicki for this article. Contact her at l.cameron@computer.org if you would like to contribute to a future *ComputingEdge* article on computing careers. Contact Prikladnicki at rafael.prikladnicki@pucrs.br.
CAREER OPPORTUNITIES

SR. SOFTWARE ENGINEERS. Des & dvlp self-checkout & POS applns using techs like Java/J2ee, C, C++, w/4690. Reqs BS in CS, CE or rltf field, & 5 yrs rltf exp. Sr. SAP Consultants. Sys study, gap analysis, mapping, dvlp/configure SD modules, perform UAT. Reqs BS in Bus Admin, IT or rltf field, & 5 yrs rltf exp. Exp in SAP R/3, 6.0 ECC Version on SD modules & test tool HPQC QTP. Educ may be equiv/frgn equiv. Comb of degrees accepted. Mtlpl opngs in Annapolis, MD & other client sites. No travel. May reqr reloc. Apply:Vedasoft, Inc, 801 Compass Way, Suite 218, Annapolis, MD 21401.EOE

MICROSOFT CORPORATION currently has the following openings in Redmond, WA: Sr. Business Planner- Dev. & implement business strategies for taking Azure products to market & optimizing ROI, https://jobs-microsoft.icims.com/jobs/5662/job. Sr. Software Engineer - Responsible for architecting responsive & scalable solutions for s/w apps. https://jobs-microsoft.icims.com/jobs/5663/job. Sr. Interaction Designer- Dev. UI & UX designs, prototypes &/or concepts for business productivity, entertainment or other s/w or h/w apps. https://jobs-microsoft.icims.com/jobs/5664/job. Multiple job openings are available for each of these categories. To view detailed job descriptions and minimum requirements, and to apply, visit the website address listed. EOE.

MICROSOFT CORPORATION currently has the following openings in Redmond, WA: Sr. Business Planner- Dev. & implement business strategies for taking Azure products to market & optimizing ROI, https://jobs-microsoft.icims.com/jobs/5662/job. Sr. Software Engineer - Responsible for architecting responsive & scalable solutions for s/w apps. https://jobs-microsoft.icims.com/jobs/5663/job. Sr. Interaction Designer- Dev. UI & UX designs, prototypes &/or concepts for business productivity, entertainment or other s/w or h/w apps. https://jobs-microsoft.icims.com/jobs/5664/job. Multiple job openings are available for each of these categories. To view detailed job descriptions and minimum requirements, and to apply, visit the website address listed. EOE.

CLOUDERA, INC. is recruiting for our Austin, TX office: Software Engineer: Design complex software applications. Mail resume w/job code #36228 to: Cloudera, Attn.: HR, 1001 Page Mill Rd., Bldg. 2, Palo Alto, CA 94304.

COMPUTER PROGRAMMER (Senior Clinical Database Programmer) (Cambridge, MA) For pharmaceutical/biotechnology applications, utilize Oracle Central Designer tool (C#, Dot Net, VB Script), INFORM, SAS/Base, SAS Macros, SAS/Graph, SAS/AF, SQL, PL/SQL, IVRS, IRT, CDA/S, & SDTM to create, modify, & test code, forms, & script that allow computer applications to run; analyze user needs & dsgn software solutions; dvlp & write computer prgms to store, locate, & retrieve specific documents, data, & information; serve as an expert in clinical database technology, capabilities & functionality; dvlp database dsgn specifications/defnitions; lead, coordinate, dsgn & dvlp clinical databases; assist in database migratns including testing; write test cases & execute test cases for clinical studies; document test results; build complex SAS datasets from clinical database; dvlp SAS Macros, templates & utilities for data reporting; utilize SDTM guidelines for building datasets; write SAS Prgrms to generate tables,

TEMASEK RESEARCH FELLOWSHIP (TRF)

A globally connected cosmopolitan city, Singapore provides a supportive environment for a vibrant research culture. Its universities Nanyang Technological University (NTU), National University of Singapore (NUS) and Singapore University of Technology and Design (SUTD) invite outstanding young researchers to apply for the prestigious TRF awards.

Under the TRF scheme, selected young researchers with a PhD degree have an opportunity to conduct and lead defence-related research. It offers:

- A 3-year research grant of up to S$1 million commensurate with the scope of work, with an option to extend for another 3 years
- Postdoctoral or tenure-track appointment (eligibility for tenure-track will be determined by the university)
- Attractive and competitive remuneration

Fellows may lead, conduct research and publish in these areas:

- Adaptive Camouflage Techniques and Technologies
- Cyber Security for Autonomous Systems
- Perception under Adverse Conditions for UGV Navigation

For more information and application procedure, please visit:

NUS – http://www.nus.edu.sg/dpr/InfoForResearchers/trf.html
SUTD – http://temasek-labs.sutd.edu.sg/funding-opportunities/temasek-research-fellowship-trf

Closing date: 23 September 2016 (Friday)

Shortlisted candidates will be invited to Singapore to present their research plans, meet local researchers and identify potential collaborators in February/ March 2017.
listings, figures & analysis of datasets; dvlp SAS coding & table templates for preparing, processing & analyzing clinical data; establish monitoring of data transfers for ongoing trials to identify study conduct or data quality issues; & ensure Clinical Study Databases are integration ready. Must have Bachelor’s deg in Comp Sci, Engg, Mechanical Engg or Mathematics, 6 yrs of exp in the job offd & exp with dvlpmt, validation, execution, maintenance, documentation, & archival of clinical data targeted for regulatory submission. Email resume to Kadmon Corporation LLC, Careers@Kadmon.com, Attn: Senior Clinical Database Programmer.

**SR. ADVSR, BUS UNIT OPS** (Framingham, MA) Undrstnd cust & partner issues, asgn proper CA Mainframe Solutn. Work w/ mainframe team to map CA tech to customers’ bus drivers. Undrstnd & articulate CA’s tech vision & strategy. Communicate CA’s mainframe capabilities & vision for prods to mainframe partners. REQS: 5 yrs of exp in job &/or a rel occ. Must have exp w/provdng strategic dirctn & acting as a thought ldr for the dvlpmnt & exec of innovative mainframe partnr strategies; Collab w/sales teams to crte, maintn & exe a partner bus

---

**Oracle America, Inc.**

**TECHNOLOGY**

Oracle America, Inc.

**APPLICATIONS DEVELOPER**

positions in **Irvine, CA**.

Job duties include: Analyze, design, develop, troubleshoot and debug software programs for commercial or end-user applications. Write code, complete programming and perform testing and debugging of applications.

Apply by e-mailing resume to yasin.cengiz@oracle.com, referencing 385.16852.

Oracle supports workforce diversity.

---

**Oracle America, Inc.**

**TECHNOLOGY**

Oracle America, Inc.

**SOFTWARE DEVELOPER**

positions in **Westminster, CO**.

Job duties include: Design, develop, troubleshoot and/or test/QA software.

Apply by e-mailing resume to rob.cuthbertson@oracle.com, referencing 385.19416.

Oracle supports workforce diversity.

---

**Oracle America, Inc.**

**TECHNOLOGY**

Oracle America, Inc.

**CONSULTING TECHNICAL MANAGER**

positions in **Redwood Shores, CA**.

Job duties include: Analyze business needs to help ensure Oracle solution meets the customer’s objectives by combining industry best practices, product knowledge, and business acumen. Travel to various unanticipated sites throughout the United States required. May telecommute from home.

Apply by e-mailing resume to mark.k.lee@oracle.com, referencing 385.13276.

Oracle supports workforce diversity.

---

**Oracle America, Inc.**

**TECHNOLOGY**

Oracle America, Inc.

**APPLICATIONS DEVELOPER**

positions in **Atlanta, GA**.

Job duties include: Analyze, design, develop, troubleshoot and debug software programs for commercial or end-user applications.

Apply by e-mailing resume to subrahmnyam.kolahala@oracle.com, referencing 385.18439.

Oracle supports workforce diversity.
plan fr mainframe prod. based on opp, cust need & strategic/dir; Adptng goals fr
bus func based on strategy changes & ensurng changes are planned & exec'd
in alignment w/ desired bus outcomes; Defng & bdng Route-to-Market (RTM)
business plans & perf mssrgn frmwrks, geo partner prgrms & through-partner mktng
strategies; Reprssntng mainframe in industry grp's & events & creating intrnl & extrnl
awareness; Prdng pipeline forecasts & bus health checks to execute sales
mgmt & partners; Bldg & mntng relshps w/partners at all levels & dvlpng/imprvng
tools, processes & procedures to incr partner efficiency; Natl internat'l travel
req approx 35% of the time. 40 hours/week; M-F; 8:30 am-5:30 pm; Send re-
sume to: Althea Wilson, CA Technologies, 201 N Franklin Street, Suite 2200, Tampa,
FL US 33602 Refer to Requisition #131921

AITA CONSULTING SERVICES require BI Developer: Design and develop OBIEE
user interface objects(Reports,Dashboards) and OBIEE metadata/logical data
models. Required to Analyze, design, de-
velop, test, maintenance and production
support. Knowledge of performance tuning
of OBIEE reporting and dashboards and ETL tools and processes. Knowledge
of SQL and PL/SQL. Requires Masters Degree in Computer Science or related.
Must be willing to travel and relocate
to Unanticipated Client Locations in the
U.S.A. Please send the resume to AITA
Consulting Services: Ref: BI0516, 6-80
Towne Center Drive, North Brunswick,
NJ 08902.

APPLICATION SUSTAIN ENGINEER, General Motors Company, Detroit,
MI. Use & improve J2EE &Web Ser-
vice-based app soft to analyze &per-
form OTA Reflash & data collection of
OnStar Electronic Control Units (ECUs) in
vehicles communicating through Vehicle
Computer Platform (VCP). Evaluate &im-
prove enterprise app architectures such as
J2EE & Web Services delivering Inter-
active Voice Response (IVR) systems,
Web Portals, Android & iPhone mobile
devices, & in-vehicle Telematics systems,
such as to support remote stop & start of
vehicles. Supervise 1 App Sustain Engr
& 5 contractual IT analysts. Automate
the deployment of VSMS (Vehicle Ser-
vices Mgmt Sys) app using Jenkins & Zuul.
Create base Linux based Operating Sys
image that can be easily deployed across
multiple servers using SALT. Master, En-
gineering or Computer Science. 36 mos
exp as Systems Analyst, IT Analyst or Engineer. Will accept bachelor or for-

en equivalent degree in Engineering or Computer Science, followed by at
least 5 years of progressive exp in the specialty, in lieu of required education & experience. Will also accept any equally
suitable combination of education, training, and/or exp which would qualify an
applicant to perform the job offered. The
required exp. must include using & im-
proving J2EE & Web Services-based app
soft to analyze & perform OTA Reflash
& data collection of OnStar ECUs in vehi-
cles communicating through VCP. Mail
resume to Alicia Scott-Wears, GM Global
Mobility, 300 Renaissance Center, Mail
Code 482-C32-D44, Detroit, MI 48265,
Ref#2453.

SR. MANAGER. Job location: Miami, FL
& any other unanticipated locations in
U.S. Travel Required. Duties: Lead Hype-
rior DRM team for projects. Design, test
& implement changes in environ, using
ERPs, Hyperion, Oracle, ODI & Automic.
Develop, maintain & enhance integra-
tion process that imports hierarchies
from enterprise sources into DRM. De-
velop integrations from DRM to multiple
ERPs. Create batch scripts to automate

SOFTWARE
Oracle America, Inc.
has openings for
SOFTWARE
DEVELOPER
positions in West Conshohocken, PA.
Job duties include: Design, develop,
troubleshoot and/or test/QA
software. May telecommute from
home.
Apply by e-mailing resume to
chad.walkowiak@oracle.com,
referencing 385.17771.
Oracle supports workforce diversity.

TECHNICAL
Oracle America, Inc.
has openings for
TECHNICAL
ANALYSTS
positions in Colorado Springs, CO.
Job duties include: Analyze user require-
ments to develop, implement, and/or
support Oracle's global infrastructure. As a
member of the IT organization, assist with
the design, development, modifications,
debugging, and evaluation of programs
for use in internal systems within a
specific function area.
Apply by e-mailing resume to
drew.chiaiese@oracle.com,
referencing req #385.17939.
Oracle supports workforce diversity.

QA ANALYST
Oracle America, Inc.
has openings for
QA
ANALYST
positions in Westborough, MA.
Job duties include: Analyze user require-
ments to develop, implement, and support Oracle's global infrastruc-
ture.
Apply by e-mailing resume to
chris.ashton@oracle.com,
referencing 385.18202.
Oracle supports workforce diversity.
process. Perform DRM requirements & doc. gathering related to hierarchies. Perform gap analysis & prep. doc. Perform impact analysis & plan for regression testing on existing Essbase cubes. Design & develop DRM hierarchies. Integrate DRM hierarchies from DRM to DRM Hub. Prep. functional design docs, tech. design docs., gap analysis docs. & interface & integration docs. Requires: M.S. in Comp. Sci., Eng. or related field & 3 yrs. exp. in job offered or 3 yrs. exp. as a Consultant, Systems Analyst or Associate. Will accept B.S. (or foreign equiv) & 5 yrs. exp. in the comp. ind. in lieu of M.S. & 3 yrs. exp. Concurrent exp. must incl.: 3 yrs. exp. creating batch scripts to automate processes; & 3 yrs. exp. designing & developing DRM hierarchies. Send resume (no calls) to: Michelle Ramirez, The Hackett Group, Inc., 1001 Brickell Bay Dr., Suite 3000, Miami, FL 33131.

SOFTWARE ENGINEER: Analysis of user requirements, design & development of custom software applications including design and full software life cycle implementation of large scale Object oriented distributed systems, ERP & EAI applications. Minimum 1 year exp. with PeopleSoft HRMS v8.1, People Tools v8.1, SQR, PeopleCode, Oracle 9i SQL Server 2005/2000 as Programmer/Analyst or any related position. MS in Computer science is required. BS or any foreign degree equivalent to BS plus 5 years of exp can be replaced for MS degree. Position requires travel to client locations as required. Relocation/Telecommuting option possible. Mail resume to: eVantage Solutions Inc, Attn: HR, 3, Independence way, Ste 209 Princeton, NJ 08540

SOFTWARE DEVELOPER: Design, develop, test and implement s/w using knowledge in Java, C++, JNL, Android SDK, NDK, ADB, DDMS, Logcat, Traceview, GIT, JIRA and Agile Development Lifecycle. Must be willing to travel & relocate to unanticipated client locations throughout the US. Req's MS in comp sci, eng or rel. Mail resumes to Informatic Technologies Inc. 900 Oak Tree Ave, Ste C2, South Plainfield, NJ 07080.

Ericsson Inc. has multiple openings for the following positions: BUSINESS CONSULTANT _ in PLANO, TX to work on consulting operations & assist the project lead in management & delivery of strategy & implementation of projects. Frequent domestic/international travel required. Job ID: 16-TX-1967. BUSINESS ANALYST _ in PLANO, TX to build business intelligence solutions & liaise with technical teams to facilitate project development. Job ID: 16-TX-2012. BUSINESS OPERATIONS ANALYST _ in PLANO, TX to coordinate the design, development and maintenance of ongoing Database integration projects. Job ID: 16-TX-166. TECHNICAL SUPPORT ENGINEER _ in BELLEVUE, WA to perform day to day Telecom Order management, analyze production issues and provide issue resolution. Job ID: 16-WA-3660. To apply please mail resume to Ericsson Inc. 6300 Legacy Drive, R1-C12 Plano, TX 75024 and indicate appropriate Job ID.
PERCEPTION SYSTEMS ENGINEER, General Motors Company, Warren, MI. Formulate, dvlp, implement & test algorithms in C++ & Python to determine maneuverable space for autonomous/self-driving vehicles by fusing info from map data, visual odometry, lane info, GPS localization & lane level path plan. Validate algorithms according to internal active collision safety standards & reqmts. Dvlp multi-modal Deep Learning systems for the Artificial Intelligence of autonomous vehicles. Master, Robotics or Robotics Systems Development. 3 mos exp as Teaching Asst or Researcher developing, testing & validating algorithms in C++ & Python to determine maneuverable space for autonomous vehicles or autonomous mobile robots by fusing info from map data, visual odometry, localization & implementing algorithms in computer hardware according to internal active collision safety standards & reqmts & developing & implementing multi-modal Deep Learning systems for artificial intelligence of autonomous/self-driving vehicles. Mail resume to Alicia Scott-Wears, GM Global Mobility, 300 Renaissance Center, Mail Code 482-C32-D44, Detroit, MI 48265, Ref#8240.

RESEARCHER, MANUFACTURING FLEXIBILITY, General Motors Company, Warren, MI. Conduct scientific research in flexible mfg systems for high volume automated production, incldg robotics, artificial intelligence, & machine learning. Research & dvlp autonomous motion planning algorithms for mfg using flexible robots such as continuum robots & hyper-redundant manipulators. Use C++, MATLAB, & tools such as OPENGL, OPENCV & Computational Geometry Algorithm Library (CGAL). Ph.D., Computer Science or Engineering. 6 mos exp as Researcher or Graduate Research Asst researching & developing autonomous motion planning algorithms for mfg using flexible robots such as continuum robots & hyper-redundant manipulators. Mail resume to Alicia Scott-Wears, GM Global Mobility, 300 Renaissance Center, Mail Code 482-C32-D44, Detroit, MI 48265, Ref#2436.

INFORMATION TECHNOLOGY BUSINESS ANALYST, General Motors Company, Detroit, MI. Define & implement Vehicle Services Management System (VSM) Product backlog ( Reqmts) using CALIBRE RM tool, to Reflash internal vehicle control modules such as OnStar control module, external control modules such as climate control module. Re-prioritize Product backlog based on customer feedback, market changes & business priorities using JIRA & Confluence. Dvlp, improve & support features for In-Vehicle Voice Messaging (IVVM) using J2EE & Java, creating vehicle data triggers & voice messaging svcs. Support new mobile app reqmts such as ignition on & remote unlock data points for mobile apps using Layer7, SOAP & XML. Master, Computer Science, Computer Applications or Information Systems. 12 mos exp as IT Analyst improving & supporting features for IVVM using J2EE & Java, creating vehicle data triggers & voice messaging svcs. & developing new mobile app reqmts & data points for the mobile apps using Layer7, SOAP & XML. Mail resume to Alicia Scott-Wears, GM Global Mobility, 300 Renaissance Center, Mail Code 482-C32-D44, Detroit, MI 48265, Ref#2701.

SENIOR SOFTWARE TEST ENGINEER, General Motors Company, Detroit, MI. Review, provide engfr feedback, & improve Service Oriented Architecture (SOA) designs & business narratives, & perform end-to-end (E2E) integration testing of In Vehicle Infotainment Systems & OnStar telematics soft & company website, collaborating with Cyber Security Team to identify & create security test scenarios & find cyber security defects, formulating & planning test strategies, performing manual & automated testing in accordance with industry standards & Agile Methodology, using UFT & QTP, Selenium & Java. Of back office systems such as Global Advisor, OnStar RemoTelLink (mobile app) & Cloud-based Infotainment systems using Sales Force cloud platform. Validate Network security for Remote Access / VPN & Intrusion Detection / Prevention using Wire Shark. Validate TCP/IP protocol ste & Operating Sys (Android). Perform Cross Browser testing of websites & APIs using protocols such as JSON & REST. Master, Computer Science & Engineering or Electrical Engineering. 36 mos exp as Engr., Lead Engr or Sr. Software Engr. Will accept bachelor or foreign equivalent, in Computer Science & Engineering or Electrical Engineering, followed by at least 5 yrs progressive exp in specialty. Will accept any equally suitable combination of education, training, &/or experience which would qualify applicant to perform job offered. Required exp must include reviewing & improving SOA designs & business narratives, & performing E2E integration testing of systems soft & company website, collaborating with Cyber Security Team, to identify & create security test scenarios & find cyber security defects, & perform automated testing using HP UFT / QTP & Selenium of back office systems, mobile apps & cloud platforms. Mail resume to Alicia Scott-Wears, GM Global Mobility, 300 Renaissance Center, Mail Code 482-C32-D44, Detroit, MI 48265, Ref#1551.

SOFTWARE DEVELOPER. Applications (Amityville, NY) sought by Alarm Lock Systems, LLC., to perform design, dvlp, review & analyze, re-engr, & integrate svrware apps to co.'s electronic & mechanical access & egress control keyless entry products. Dvlp svrware apps to security access systms; re-structure & up-grade legacy svrware systms. Perform test & debug industry specific svrware for lock systms. Min. req.: BSc. in Comp Sci, must incl coursework in Algorithm Dsgn & Analysis, Operating Systms, Compiler Dsgn, & Operating Systms, + 2-yr in-job exp. &/or 4-yr in-job exp. &/or 4 yrs in svrware. Must be proficient in WCF, WPF, XML, multi-threading, sockets in high volume & sensitive data envrmt., & SQL Server dbase, & ASP.NET 3.5/4/5, ASP.NET MVC4/5, TCP/IP, Communication, MS SQL Server. Mail resume to Louis Mollica, Recruiter, Alarm Lock Systems LLC., 333 Bayview Ave, Amityville, NY 11701. EOE. No calls/walk-ins.

CLOUDERA, INC. is recruiting for our Austin, TX office: Software Engineer: design & implement distributed systems that scale well – to petabytes of data and thousands of nodes. Mail resume w/job code #36230 to: Cloudera, Attn.: HR, 1001 Page Mill Rd., Bldg. 2, Palo Alto, CA 94304.

CLOUDERA, INC. is recruiting for our Palo Alto, CA office: Software Engineer: Utilize Java and Python to build distributed systems that just work from a functionality, performance and scalability standpoint. Mail resume w/job code #37126 to: Cloudera, Attn.: HR, 1001 Page Mill Rd., Bldg. 2, Palo Alto, CA 94304.
SOFTWARE
Oracle America, Inc. has openings for
SOFTWARE DEVELOPER
positions in Bedford, MA.

Job duties include: Design, develop, troubleshoot and/or test/QA software.

Apply by e-mailing resume to huimin.xu@oracle.com, referencing 385.18524.

Oracle supports workforce diversity.

QA ANALYSTS
Oracle America, Inc. has openings for
QA ANALYST
positions in Bedford, MA.

Job duties include: Design, install and evaluate quality control methods and systems. Develop standards and procedures to provide quality guidance and methods. Perform manual and automated testing of next generation session border controllers.

Apply by e-mailing resume to shane.clark@oracle.com, referencing 385.17602.

Oracle supports workforce diversity.

TECHNOLOGY
Oracle America, Inc. has openings for
DATABASE ADMINISTRATOR
positions in Frisco, TX.

Job duties include: Provide enterprise-wide, database administration support for production systems and provide DBA services to application development teams, including database design, database generation, coding, and/or database production support.

Apply by e-mailing resume to walter.lee@oracle.com, referencing 385.17769.

Oracle supports workforce diversity.

TECHNOLOGY
Oracle America, Inc. has openings for
APPLICATIONS DEVELOPER
positions in Solon, OH.

Job duties include: Analyze, design, develop, troubleshoot and debug software programs for commercial or end-user applications. Write code, complete programming and perform testing and debugging of applications.

Apply by e-mailing resume to matt.shields@oracle.com, referencing 385.18245.

Oracle supports workforce diversity.

QA ANALYSTS
Oracle America, Inc. has openings for
QA ANALYST
positions in Bedford, MA.

Job duties include: Design, install and evaluate quality control methods and systems. Develop standards and procedures to provide quality guidance and methods. Perform manual and automated testing of next generation session border controllers.

Apply by e-mailing resume to shane.clark@oracle.com, referencing 385.17081.

Oracle supports workforce diversity.

SOFTWARE
Oracle America, Inc. has openings for
SOFTWARE DEVELOPER
positions in Burlington, MA.

Job duties include: Design, develop, troubleshoot and/or test/QA software. As a member of the software engineering division, apply knowledge of software architecture to perform tasks associated with developing, debugging, or designing software applications or operating systems according to provided design specifications.

Apply by e-mailing resume to dipankar.bajpai@oracle.com, referencing 385.15700.

Oracle supports workforce diversity.
Oracle America, Inc. has openings for

CONSORTING TECHNICAL MANAGER
positions in Sunrise, FL.

Job duties include: Analyze business needs to help ensure that the solution meets the customer's objectives by combining industry best practices, product knowledge, and business acumen. Travel to various unanticipated sites throughout the United States required.

Apply by e-mailing resume to brian.shumsky@oracle.com, referencing 385.16588.

Oracle supports workforce diversity.

Oracle America, Inc. has openings for

MANAGING PRINCIPLE CONSULTANT
positions in Redwood Shores, CA.

Job duties include: Responsible for providing quality work products on customer engagements while managing a small team of consultants. Utilize broad understanding of solutions, industry best practices, multiple business processes or technology designs within a product/technology family. Travel to various unanticipated sites throughout the United States required. May telecommute from home.

Apply by e-mailing resume to ricardo.m.martinez@oracle.com, referencing 385.18485.

Oracle supports workforce diversity.

Oracle America, Inc. has openings for

DATABASE ADMINISTRATOR
positions in Frisco, TX.

Job duties include: Provide enterprise-wide, database administration support for production systems and provide DBA services to application development teams, including database design, database generation, coding, and/or database production support. Offer DBA support with a high degree of customer service, technical expertise, and timeliness.

Apply by e-mailing resume to walter.l.lee@oracle.com, referencing 385.17992.

Oracle supports workforce diversity.

Oracle America, Inc. has openings for

SOFTWARE DEVELOPER
positions in Marlborough, MA.

Job duties include: Design, develop, troubleshoot and/or test/QA software.

Apply by e-mailing resume to piriyakala.suresh@oracle.com, referencing 385.16930.

Oracle supports workforce diversity.

Oracle America, Inc. has openings for

RELEASE DEVELOPER
positions in Deerfield, IL.

Job duties include: Develop, analyze and maintain tools that support and automate processes for hardware or software product release.

Apply by e-mailing resume to Richard.L.Jones@oracle.com, referencing 385.18605.

Oracle supports workforce diversity.
Juniper Networks is recruiting for our Sunnyvale, CA office:

**Test Engineer #33059**: Work alongside technical leads on critical aspects of the security software testing. Test complex multi-service networking & network security products for Data Center & cloud.

**Technical Support Engineer #40043**: Provide tier 3 technical support services to Company’s end-user enterprise, carrier, and service provider customers, and channel partners.

**Software Engineer Staff #8878**: Develop software forwarding ASICs, such as Broadcom and Marvell. Design and develop kernel drivers and applications for operating systems.

**Distinguished Engineer #34053**: Design and develop software and solutions for datacenter network orchestration and management. Present Juniper’s Contrail Networking solution to customers and at summits.

**Software Engineer #12202**: Design, develop, troubleshoot and debug the packet forwarding path for Layer-2/Layer-3 switches. Write drivers for custom ASICs and FPGAs, network processors and Ethernet switches.

**QA Engineer Staff #38706**: Test complex multi-service networking and network security products for Data Center and cloud. Test L4-L7 security infrastructure, SLL proxy, web security, app security, and app service components.

**Software Engineer Staff #17941**: Perform software design and development of company products. Design, develop and maintain control and UI modules for company’s SDN platform.

**Technical Support Engineer #37649**: Provide high-level expertise on company specific products. Deliver in-depth diagnostics & root-cause analysis for network impacting issues on Juniper’s routing products to large internet service providers & enterprise customers.

**QA Engineer Staff #11057**: Write test plans, design test cases, and perform manual and automated testing of Juniper’s Contrail cloud platform features and Contrail Networking products.

**Juniper Networks is recruiting for our Westford, MA office:**

**Technical Support Engineer #34006**: Provide technical support to customers to resolve problems related to routers, protocols and network design.

**QA Engineer #30732**: Develop comprehensive test plans based on changing and challenging product definitions, scaling and performing targets, and customer use case scenarios.

**Hardware Engineer #37447**: Participate in hardware design and development from concept through final release. Ensure that EMC compliance is designed into products.

**Technical Support Engineer #28858**: Be the focal technical support contact & handle high-priority issues to ensure prompt restoration and resolutions.

**QA Engineer #12642**: Execute test cases for qualifying Juniper SDN product portfolio. Understand customer deployment topologies involving Juniper MX, QFX, and Contrail products.

**Test Engineer #32511**: Review functional specifications for testable requirements and write detailed test plans. Execute testing of protocols on Juniper hardware and document test execution pass or fail for each test case.

**Internal Auditor #27238**: Conduct internal information technology system and SOX audits and SOC reports review, and support Financial and Integrated Audit team with determining the reliability and effectiveness of internal control systems.

**Software Engineer #23083**: Design, develop, and debug code to program the forwarding ASIC to perform various functionalities. Code in C languages and use GDB and debugging tools.

**Software Engineer #16532**: Design, develop, administer, troubleshoot, and maintain software for the manual and automated test environment for Automated Support and Prevention (ASAP) products.

**Technical Support Engineer #37547**: Provide technical assistance & conduct troubleshooting to resolve hardware/software issues. Identify service support initiatives to improve the supportability of processes, products & systems.

**Technical Support Engineer #24949**: Work with customers to resolve issues related to router, protocols, and network design. Troubleshoot h/w and s/w issues, and replicate customer environment and network problems in the lab.

---

**Mail single-sided resume with job code # to**

Juniper Networks
Attn: MS A.8.429A
1133 Innovation Way
Sunnyvale, CA 94089
**CAREER OPPORTUNITIES**

**ENGINEERING**

Software Development Engineer (Req#A9UTSL) Assess & provide direction to offshore developers.

Hardware Development Engineer (Req#0DCTG2) Dsgn & dvp HW for mobile devices. Travel req. 15%.

Software Development Engineer (Req#9SV6ZQ) Dsgn, Implmnt & deploy large scale srch srvcs.

Tooling Engineer (Req#9SYS2T) Participate in the prdct des process through appropriate Design for Manufacturing (DFM) input in metal components. Travel Req’d 30%.

Software Engineer Applications (Req#9LLNRZ) Res for the overall health & quality of SW apps.

ASIC Design Engineer (Req#9PG42C) Explore adv tech & dev packaging solutions that are targeted for high vol manufcting (HVM). Travel Req’d 15%.

Software Development Engineer (Req#A2P2S7) Dev, des, & impl architecture for SW components.

Hardware Development Engineer (Req#9FU29K) Develop processes & tech for organic light-emitting diode (OLED) displays. Travel req. 25%.

Senior Network Engineer (Req#9RMNKK) Build & maintain Apple’s Data Center Network Infrastructure.

Software Engineer Applications (Req#A6336) Architect, author & deliver SW to improve the avail, scal & secure of Apple’s internet services.

Human Interface Designer (Req#-A2SVZV) Dsgn & dvp innovative & compelling interactive prototypes by taking new concepts & building them into something people can see, touch, & experience.

Software Development Engineer (Req#9EPLXH) Embed SW des & dev on base & iPhone app process.

ASIC Design Engineer (Req#9K2PJW) Compse a pre-Silicon verification plan of a HW unit or subsys based on dsgn requirmnts & micro-architctre.

Software Engineer Applications (Req#9X-JVWF) Manage Maps svcs in prod environs & provide the des, suppv & automation of business sys critical for the Maps org to build a great user exp.

Software Development Engineer (Req#9YG2XW) (Multiple Positions Open) Des, debug, & bring-up next gen iOS devcs.

Software Development Engineer (Req#9UK487) Rsrch, des, dev & debug front-end compiler tech.

Software Engineer Applications (Req#9XLV7) Des, dev, test & debug SW apps for Apple Retail Store POS.

Software Engineer Applications (Req#9TE333) Dsgn, dev, & test custmr support web features/functions delivered via apps & svcs.

Data Mining Specialist (Req#9U95C8) Dev sys of automdl prgrammng modules for scaled txt data trnsfr & analysis.

Hardware Development Manager (Req#9ESN42) Supp HW des tm respon-sib for des & dev camera modules for consumer elect devices. Travel req’d 20%.

Hardware Development Engineer (Req#-9CYTMY) Dsgn & execute tests of elec compnents of camera moduls.

Software Engineer Applications (Req#9FMNW5) Dsgn & dev prodctvty app w a focus on srvr side tech & client/ server intrction.

Software Development Engineer (Req#9W87TF3) Dsgn big data sys to store info for tools used in maps orgnztn. Model data for NoSQL & stdnd Rdbs.

Engineering Systems Administrator (Req#9UJSSE) Respons for site reliability for network attached storage systems.

Software Engineer Systems (Req#A3939A) Des, dev & mntn high perf & scalable dist svcs that form the core of Cust Syst Platform.

ASIC Design Engineer (Req#9LTVFY) Dvp & implmnt dsgn for testability (DFT) & svcs.

Senior Project Manager (Req#9QYTKR) Supp the open’s & remodels of Apple Retail Stores globally, in accord w Apple stdns.

Firmware Engineer (Req#9V4UF9) Dev & tune algorithms for processing sig-nals from next-gen sensors

Data Scientist (Req#9DCM88) Dev & perform in-depth analysis, modeling, & visualization of unstructured data for eng des teams.

Software Development Engineer (Req#9HJNHE) Des & dev kernel SW, sys Sv, & tools for perf analysis.

Software Engineer Applications (Req#9XEE27) Des, implmnt & secure SW for Apple Retail worldwide.

Engineering Project Manager (Req#9H3UT8) Coordinate display prog teams for iPad prdcts. Dsgn & track prog scheds & milestones.

Engineering Project Coordinator (Req#9R6TFD) Dvlp & coordinate new display tech for mobile devices.

Systems Design Engineer (Req#9U3QKT) Prfrm Radio Frequency (RF) systm dsgn validation & debug for wireless telecom sysms.

Software Development Engineer (Req#A2M2SY) Des & implmnt OS SW framwrk for media streaming.

Software Development Engineer (Req#A87TBW) Bridge the gap btw SW eng team & expert users to optimize the usage & benefits from SW sys.

Engineering Program Administrator (Req#9UQTA) Coordinate new display tech for mobile devices. Travel Req 30%.

Engineering Project Coordinator (Req#9KYPN2) Dsgn & dlvp SW apps for Apple’s service mgmt. space to spprt Retail & Partner channel sysms.

Hardware Development Engineer (Req#9EZW3R) Des, dev, inotr, charac & valid sensor sys for consmrl ecel pdos. Travel Req 25%.

Systems Design Engineer (Req#9JWTTA) Eval latest iPad, iPhone & Apple Watch HW sys. Travel req 25%.

Software Quality Assurance Engineer (Req#9WG2K9) Dev test scripts to test app prfmrnc, scalblty & reliability.

Software Systems Engineer (Req#9V5Q9C) Des & dev SW for high perfmcve GPU rendering.

Software Quality Assurance Engineer (Req#9USU29) Des & dev QA automat framwrk for testing web services.

Software Development Engineer (Req#9WXRK) Des & dev SW for image proc’ng sys. Alyze, resch, & eval algorthm models for image proc’ng & comp vision.

Software Engineer Applications (Req#9S-9V6D) Build web based svl svc solns that auto provide resources like load balancers & firewalls rel to datacenter mgmt.

Software Development Engineer (Req#9WZ342) Des & dev Siri backend
CAREER OPPORTUNITIES

Apple Inc. has the following job opportunities in Austin, TX:

ASIC Design Engineer (Req#9PUHY)
Work w/ micro-arch to define memory subsys, perform feasibility, make area/frequency/performance/power tradeoffs, & des & balance pipeline stages.

ASIC Design Engineer (Req#9LE2XX)
Verify microprocessor functionality in Central Processing Units (CPUs).

ASIC Design Engineer (Req#9E4TEJ)
Complete phys implementation of complex SOC blocks.

Apple Inc. has the following job opportunities in Orlando, FL:

ASIC Design Engineer (Req#9HPZR)
Dsgn & dev HW for Graphics Processing Units.

Refer to Req# & mail resume to Apple Inc., ATTN: L.J., 1 Infinite Loop 104-1GM, Cupertino, CA 95014. Apple is an EOEE/AA m/f/disability/vets.

www.computer.org/computingedge
Watch the World’s Leading Experts Take Multi-Core Strategies to New Heights

Purchase the IEEE Computer Society advanced Multi-Core lecture series and receive a complimentary SWEBOK Knowledge Area review course of your choice.

Please visit our website for more information about this outstanding Multi-Core Video Series.

http://www.computer.org/web/education/multicore-video-series
CiSE is the perfect combination

At the intersection of science, engineering, and computer science, Computing in Science & Engineering (CiSE) magazine is where conversations start and innovations happen. CiSE appears in the IEEE Xplore and AIP library packages, representing more than 50 scientific and engineering societies.
In 1982, on the occasion of its thirtieth anniversary, the IEEE Computer Society established the Computer Entrepreneur Award to recognize and honor the technical managers and entrepreneurial leaders who are responsible for the growth of some segment of the computer industry. The efforts must have taken place over fifteen years earlier, and the industry effects must be generally and openly visible.

The award is a museum-quality sterling silver chalice commissioned from Washington DC artist Michael Schwartz. The gold-plated crown below the cup displays the alternating symbols of the IEEE and the Computer Society, supported by clusters of laurel leaves, an ancient symbol for outstanding achievement.

All members of the profession are invited to nominate a colleague who they consider most eligible to be considered for this award. Awarded to individuals whose entrepreneurial leadership is responsible for the growth of some segment of the computer industry.

**DEADLINE FOR 2017 AWARD NOMINATIONS**

DUE: 15 OCTOBER 2016

- The award nomination requires a minimum of 3 endorsements
- Presented at the highly acclaimed IEEE-CS Annual Awards Ceremony.

**Past Recipients Include:** Diane Greene, Mendel Rosenblum, Sandy Bosack, John Warnock, Charles Geschke, Michael Dell.
Explore these Software Resources

Build Your Knowledge

*IEEE Software*

The authority on translating software theory into practice, this bimonthly magazine presents pioneering ideas, expert analyses, and thoughtful insights to help software professionals keep up with rapid technology change.

*IEEE Transactions on Software Engineering*

This bimonthly journal focuses on well-defined theoretical results and empirical studies with real potential to impact software construction, analysis, and management.

Advance Your Career

**Software Engineering Knowledge Areas**

The 12 certificate areas assess a candidate’s proficiency in understanding the concepts involved, as well as their ability to apply them in the software engineering process.

- Requirements
- Design
- Construction
- Testing
- Maintenance
- Configuration Management
- Engineering Management
- Engineering Process
- Engineering Models and Methods
- Quality
- Engineering Economics

**Software Quality and Maintenance Associate Engineer Certification**

This certification is designed to recognize candidates who have gained the basic knowledge and understanding required for developing software products. It demands a coherent and demonstrable understanding of the principles and processes involved in software requirements, software design, software construction, and software testing.

FOR DIRECT LINKS TO THESE RESOURCES, VISIT

www.computer.org/edge-resources
IEEE Computer Society’s Conference Publishing Services (CPS) is now offering conference program mobile apps! Let your attendees have their conference schedule, conference information, and paper listings in the palm of their hands.

The conference program mobile app works for Android devices, iPhone, iPad, and the Kindle Fire.

For more information please contact cps@computer.org