COMPUTING edge

SOFTWARE

Also in this issue:

> Next-Generation Mobile Services

> Can Blockchain Strengthen the Internet of Things?

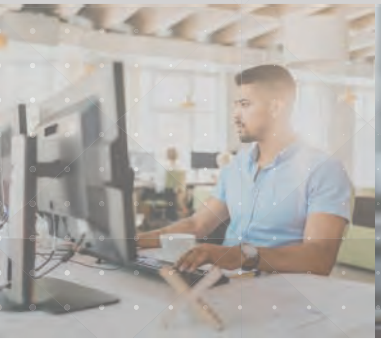
DECEMBER 2017

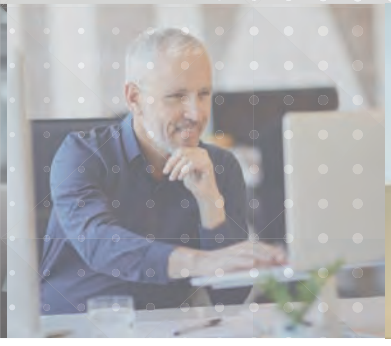
www.computer.org

IEEE

IEEE computer society

PREPARE TO CONNECT






The IEEE Computer Society is launching **INTERFACE**, a new communication tool to help members engage, collaborate and stay current on CS activities. Use **INTERFACE** to learn about member accomplishments and find out how your peers are changing the world with technology.

We're putting our professional section and student branch chapters in the spotlight, sharing their recent activities and giving leaders a window into how chapters around the globe meet member expectations. Plus, **INTERFACE** will keep you informed on CS activities so you never miss a meeting, career development opportunity or important industry update.

Launching this spring. Watch your email for its debut.

IEEE computer society

INTERFACE

COMPUTING edge

IEEE COMPUTER SOCIETY http://computer.org • +1 714 821 8380

Printed with inks containing soy and/or vegetable oils

IEEE computer society

STAFF

Circulation: ComputingEdge (ISSN 2469-7087) is published monthly by the IEEE Computer Society. IEEE Headquarters, Three Park Avenue, 17th Floor, New York, NY 10016-5997; IEEE Computer Society Publications Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; voice +1 714 821 8380; fax +1 714 821 4010; IEEE Computer Society Headquarters, 2001 L Street NW, Suite 700, Washington, DC 20036.

Postmaster: Send address changes to ComputingEdge-IEEE Membership Processing Dept., 445 Hoes Lane, Piscataway, NJ 08855. Periodicals Postage Paid at New York, New York, and at additional mailing offices. Printed in USA.

IEEE Computer Society Magazine Editors in Chief

Computer
Sumi Helal, *Lancaster University*

IEEE Software
Diomidis Spinellis, *Athens University of Economics and Business*

IEEE Internet Computing
M. Brian Blake, *University of Miami*

IT Professional
San Murugesan, *BRITE Professional Services*

IEEE Security & Privacy
Ahmad-Reza Sadeghi, *Technical University of Darmstadt*

IEEE Micro
Lieven Eeckhout, *Ghent University*

IEEE Computer Graphics and Applications
L. Miguel Encarnação, *ACT, Inc.*

IEEE Pervasive Computing
Maria Ebling, *IBM T.J. Watson Research Center*

Computing in Science & Engineering
Jim X. Chen, *George Mason University*

IEEE Intelligent Systems
V.S. Subrahmanian, *University of Maryland*

IEEE MultiMedia
Yong Rui, *Lenovo Research and Technology*

IEEE Annals of the History of Computing
Nathan Ensmenger, *Indiana University Bloomington*

IEEE Cloud Computing
Mazin Yousif, *T-Systems International*

DECEMBER 2017 • VOLUME 3, NUMBER 12

COMPUTING edge

12 Software Engineering

14 Security Challenges and Opportunities of Software-Defined Networking

38 Superhuman Sports: Applying Human Augmentation to Physical Exercise

8 Editor's Note: The Ins and Outs of Today's Software Technology

9 Why Software Is Like Baseball
RICARDO VALERDI

12 Software Engineering
RICK KAZMAN

14 Security Challenges and Opportunities of Software-Defined Networking
MARC C. DACIER, HARTMUT KÖNIG, RADOSLAW CWALINSKI, FRANK KARGL, AND SVEN DIETRICH

20 Software Reliability Redux
DIOMIDIS SPINELLIS

24 Softwarization of Internet of Things Infrastructure for Secure and Smart Healthcare
MOHAMMAD A. SALAHUDDIN, ALA AL-FUQAHA, MOHSEN GUIZANI, KHALED SHUAIB, AND FARAG SALLABI

30 The Design and Architecture of Microservices
ALAN SILL

36 Next-Generation Mobile Services
M. BRIAN BLAKE

38 Superhuman Sports: Applying Human Augmentation to Physical Exercise
KAI KUNZE, KOUTA MINAMIZAWA, STEPHAN LUKOSCH, MASAHIKO INAMI, AND JUN REKIMOTO

42 Can Blockchain Strengthen the Internet of Things?
NIR KSHETRI

47 Fully Autonomous Driving: Where Technology and Ethics Meet
DIETER BIRNBACHER AND WOLFGANG BIRNBACHER

50 Computer-Aided Fashion
CHARLES DAY

Departments

4 Magazine Roundup

51 Computing Careers: Careers in Software Engineering

47

Fully Autonomous Driving: Where Technology and Ethics Meet

Magazine Roundup

by Lori Cameron

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 design. Here are highlights from recent issues.

Computer

Real-Time Video Analytics: The Killer App for Edge Computing

Experts expect the number of surveillance cameras installed worldwide to grow 20 percent every year for the next five years. They are used for a variety of purposes including traffic control, surveillance, and security. The need for real-time video analytics is crucial because public-safety officials often must respond quickly to footage of accidents and criminal activity. In the October 2017 issue of *Computer*, researchers say that video analytics will drive a wide range of applications with great potential to impact society. A geographically distributed architecture of public clouds and edge networks that extend down to the cameras is the only feasible approach to meeting the strict real-time requirements of large-scale live video analytics.

Computing in Science & Engineering

Toward Exascale Earthquake Ground Motion Simulations for Near-Fault Engineering Analysis

Earthquake ground motions pose an ever-present risk to engineered structures and the infrastructure that modern life depends on. Civilization has evolved in close proximity to active earthquake faults and sedimentary basins that amplify seismic motions. However, many cities at high ground-motion risk haven't experienced motion-related damage due to long time intervals between large earthquake events. In the September/October issue of *Computing in Science & Engineering*, researchers say application modernization for massively

Published by the IEEE Computer Society

parallel time-domain simulations of earthquake ground motion in 3D models is increasing application resolution and providing ground-motion estimates for critical-infrastructure risk evaluations. Improvements to the geophysics application code SW4, developed while porting the code to systems at the US Lawrence Berkeley National Laboratory, revealed that reorganizing operation order could improve performance.

IEEE Annals of the History of Computing

IBM Branch Offices: What They Were, How They Worked, 1920s–1980s

In the 20th century, IBM opened more than 800 sales and equipment-maintenance branch offices around the world. They were scattered across nearly 170 countries and became IBM's physical footprint visible to customers and communities. During the century, they housed tens of thousands of employees, making them the largest collection of groups and buildings belonging to any company in the world of information processing. IBM's sales and customer support came out of these organizations. Read more about this in the July–September 2017 issue of *IEEE Annals of the History of Computing*.

IEEE Cloud Computing

Towards Transparent and Trustworthy Cloud

Despite its immense benefits in terms of flexibility, resource consumption, and simplified management, cloud computing raises several concerns due to a lack of trust and transparency. Like all computing paradigms based on outsourcing, the use of cloud computing is largely a matter of trust. There is increasing pressure by cloud customers for solutions that would increase their confidence that a cloud-based service or application is behaving in a secure and correct manner. In the May/June 2017 issue of *IEEE Cloud Computing*, researchers say cloud assurance techniques, developed to assess the trustworthiness of cloud services, can play a major role in building trust.

IEEE Computer Graphics and Applications

Urban Space Explorer: A Visual Analytics System for Urban Planning

Understanding people's behavior is fundamental to many planning professions—transportation, community development, economic development, and urban design—that rely on data about frequently traveled routes, places, and social and cultural practices. Based on the results of a practitioner survey, the authors of "Urban Space Explorer: A Visual Analytics System for Urban Planning," which appears in the September/October 2017 issue of *IEEE Computer Graphics and Applications*, designed Urban Space Explorer. This visual-analytics system utilizes mobile social media to enable interactive exploration of public-space-related activity along spatial, temporal, and semantic dimensions.

IEEE Intelligent Systems

Computers Play Chess, Computers Play Go *... Humans Play* Dungeons & Dragons

With the AlphaGo computer program's recent win over one of the world's expert *Go* players, AI researchers need to explore new challenges in the game-playing arena. While there are a number of games to explore, the authors pose a true challenge for the next decade: attacking human-oriented games such as *Dungeons & Dragons*. Read more in the July/August 2017 issue of *IEEE Intelligent Systems*.

IEEE Internet Computing

TCP and MP-TCP in 5G mmWave Networks

A spectrum exists between microwave and infrared waves that promises to redefine high-speed wireless communication: the millimeter wave spectrum (mmWave), which has a range of 30 to 300 GHz. These frequencies frequently experience highly dynamic channel conditions, which lead to wide fluctuations in the received signal's quality. The authors of "TCP and MP-TCP in 5G mmWave Networks," which appears in the September/October 2017 issue of *IEEE Internet Computing*, explain how the end-to-end user experience in mobile mmWave networks could be affected by poor interaction with the most widely used transport protocol: TCP. They

also provide insights into the throughput-latency tradeoff when Multipath TCP (MP-TCP) is used judiciously across various links.

IEEE Micro

BRAIN: A Low-Power Deep Search Engine for Autonomous Robots

Researchers are studying the use of autonomous robots in many unmanned applications. However, the robots' heavy costs and limited battery life make it difficult for them to implement intelligent decision making. In response, researchers propose a low-power deep search engine (code-named "BRAIN") for real-time path planning of intelligent autonomous robots. To achieve low power consumption while maintaining high performance, BRAIN adopts a multithreaded core architecture with a transposition table cache to detect and avoid duplicated searches between the processors at the deeper levels of the search tree. BRAIN achieves fast search speed and low energy consumption, while the robots navigate successfully without collision. Read more about this innovative search engine in the September/October 2017 issue of *IEEE Micro*.

IEEE MultiMedia

Augmented Reality in Reality

The use of augmented reality (AR) in smartphones has been soaring, thanks to breakthroughs in AR algorithms. In July 2016, Niantic and Nintendo released *Pokemon Go*, triggering millions of downloads in one week. One month later, social media giant Tencent organized the virtual Olympic torch relay on smartphones, encouraging 100 million people to use AR techniques provided by HiScene. The trend became clearer in 2017, with Snapchat's release of World Lenses 7 and the popularity of Meitu's photo-editing app. Each of these apps has hundreds of millions of active users. This article surveys academic contributions driving AR's commercial potential, as well as the industry trends advancing software and hardware developments. The author offers advice on how start-ups hoping to leverage these advances can compete against established vendors. Read more in the July–September 2017 issue of *IEEE MultiMedia*.

IEEE Pervasive Computing

Sensing, Privacy, and Things We Don't Discuss

Brand new tech developments can be fascinating or just downright bizarre. Imagine the following: the ability to change the volume, filters, and sound effects of a guitar simply by touching its face; an olfactory sensor that emits scents that can improve your mood; sensors that can tell if your produce or fish is going bad; or electronic devices that dissolve in water, ensuring that your private data has been wiped out. The article "Sensing, Privacy, and Things We Don't Discuss," which appears in the July–September 2017 issue of *IEEE Pervasive Computing*, covers the development of all these technologies and more.

IEEE Security & Privacy

Security and Privacy Experiences and Practices of Survivors of Intimate Partner Abuse

Intimate-partner abuse can be a harrowing, life-threatening experience for victims. Often, unfortunately, current tech advances seem to aid the perpetrator in contacting, tracking down, harassing, and intimidating the victim. Now, researchers are seeking to turn that around. Recognizing how intimate-partner abuse's three phases—physical control, escape from the abuser, and life apart from the abuser—affect survivors' technology use, researchers can better understand and support this population's digital security and privacy needs. Read more about this in the September/October 2017 issue of *IEEE Security & Privacy*.

IEEE Software

Improving the State of Automotive Software Engineering

The automotive industry is fundamentally changing by becoming software intensive, rather than mechanically intensive. To stay ahead of the game, automakers must continuously improve their software engineering. The authors of "Improving the State of Automotive Software Engineering," which appears in the September/October 2017 issue of *IEEE Software*, studied the existing literature on the subject and made practitioner-oriented recommendations.

IT Professional

Managing Diabetes Therapy through Datastream Mining

In insulin-dependent diabetes therapy, taking the right insulin dosage at the appropriate times is essential. In the article "Managing Diabetes Therapy through Datastream Mining," which appears in the September/October 2017 issue of *IT Professional*, the authors propose a datastream mining approach that computationally derives real-time decision rules for formulating insulin-dependent diabetes therapy based on prescription records and the patient's blood-glucose reactions. These decisions are based on the patient's current health conditions, not general historical data of a population over several years. The rules thus more accurately predict whether a medical problem will occur, given that glucose levels fluctuate because of lifestyle changes, medications, or other external factors.

Computing Now

The Computing Now website (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.

EDITOR'S NOTE

The Ins and Outs of Today's Software Technology

This *ComputingEdge* issue focuses on the current state of software technology, the challenges it faces, and what the future holds.

Baseball teams use sabermetrics to make key personnel decisions. Applying this approach to software projects might help development teams operate more effectively, according to *IEEE Software*'s "Why Software Is Like Baseball."

As the importance of software in our world increases, so do the duties, skills, and knowledge required of software engineers, explains "Software Engineering," from *Computer*.

The authors of *IEEE Security & Privacy*'s "Security Challenges and Opportunities of Software-Defined Networking" discuss software-defined networking's (SDN's) security issues, strategies to monitor and protect SDN-enabled networks, and strategies for leveraging SDN in the design of new security mechanisms

Avoiding problems in a world in which high reliability is becoming necessary in an increasing number of applications won't be easy, says the author of *IEEE Software*'s "Software Reliability Redux."

"Softwarization of Internet of Things Infrastructure for Secure and Smart Healthcare," from *Computer*, proposes an agile, softwarized infrastructure for the flexible, cost-effective, and secure deployment of Internet of Things systems for smart healthcare applications and services.

Microservices—in which developers build large applications as suites of modular services that communicate with one another—is the focus of *IEEE Cloud Computing*'s "The Design and Architecture of Microservices."

Articles on topics other than software include:

- *IEEE Internet Computing*'s "Next-Generation Mobile Services" looks at possible futuristic mobile-device approaches.
- "Superhuman Sports: Applying Human Augmentation to Physical Exercise," from *IEEE Pervasive Computing*, examines an emerging research field focused on exploring a new use for human augmentation.
- The author of "Can Blockchain Strengthen the Internet of Things?" from *IT Professional*, highlights how blockchain-based solutions could be, in many ways, superior to the current Internet of Things ecosystem, which relies mainly on centralized cloud servers.
- *IEEE Intelligent Systems*' "Fully Autonomous Driving: Where Technology and Ethics Meet," argues that in designing self-driving vehicles, in which safety is critical, it's important to separate the tasks of technology and ethics, as well as the responsibilities of different stakeholders.
- The author of "Computer-Aided Fashion," from *Computing in Science & Engineering*, describes how software could help with fashion construction.

Published by the IEEE Computer Society

INVITED CONTENT

Editor: **Giuliano Antoniol**
Polytechnique Montréal
antoniol@ieee.org

Editor: **Steve Counsell**
Brunel University
steve.counsell@brunel.ac.uk

Editor: **Phillip Laplante**
Pennsylvania State University
pal11@psu.edu

Why Software Is Like Baseball

Ricardo Valerdi

THERE ARE MANY parallels between software and baseball. Besides the teamwork involved, an individual's contributions to the outcome are important in both settings. Put on your thinking cap for a moment as we explore how programmers can be evaluated the same way baseball players are. (For those of you unfamiliar with the sport, the "Baseball Basics" sidebar provides a very brief tutorial.)

Statistics, Sabermetrics, and Software

The convergence of technology and sports has gained tremendous momentum as professional leagues have found ways to connect with a new generation of fans via mobile devices. While watching or attending a baseball game, fans might toggle between social media apps such as Twitter and MLB.com At Bat to check on game-related information and statistics. (For more on MLB.com At Bat, see the related sidebar.)

One such statistic is a team's *win probability*, p(win), which is based on each play's outcome. Both teams begin a game with p(win) = 0.5, but after the first result occurs on the field, the probabilities change. For example, scoring a run will increase a baseball team's p(win) by a certain percentage while reducing the other team's p(win) by the same amount. The sum of both teams' p(win) will always equal 1.

The sports industry has heavily invested in such analytics for some time. Finding inefficiencies in the industry has given teams and coaches a slight edge over their competitors that might translate into

- more wins per million dollars (*win efficiency*) and
- additional revenue through ticket sales, corporate sponsorship, and television contracts (*franchise value*).

Professional baseball in particular has a tradition of analytics. This was documented in the book *Moneyball*,[1] which suggested that

- players should be evaluated on past performance rather than potential;
- certain metrics are overvalued, rewarding individual behavior rather than team behavior; and
- there's hidden value in recruiting often-overlooked players who value team performance rather than expensive superstars who value individual accomplishments.

Such thinking led to the formation of the baseball analytics movement, now called *sabermetrics*. Sabermetrics uses data to make objective decisions about which players to draft, which players to play, how much to pay players, and which personnel trades between teams make the most sense.

2469-7087/17/$33.00 © 2017 IEEE Published by the IEEE Computer Society December 2017

BASEBALL BASICS

In baseball, a player normally scores a *run* (a point) by hitting a pitched ball with a bat and then running from *home plate* to *first base*, to *second base*, to *third base*, and back to home plate. Each run is credited to the player who scored it.

A player's *batting average* is the ratio of hits (successful outcomes) to *at bats* (opportunities to hit) for a particular period. A hit can be a *single* (the hitter reaches first base), a *double* (the hitter reaches second base), a *triple* (the hitter reaches third base), or a *home run* (the hitter reaches home plate). A *walk* is when the batter reaches first base because a pitcher threw four balls that were out of the *strike zone* (the permitted area for pitches).

Similarly to cricket, players on each team have the opportunity to score runs during phases of play called *innings*. A professional baseball game normally comprises nine innings. Game progress is measured by the number of *outs* (usually unsuccessful at bats). A typical baseball game has 27 outs for each team (three outs per inning).

MLB.COM AT BAT

MLB.com At Bat, from Major League Baseball's (MLB's) Advanced Media division, is the top-grossing app in Apple's App Store (in the US). In terms of revenue, it consistently has been in the top 10, alongside apps such as Netflix, Pandora, and YouTube. Even more impressive, it has been the top-grossing sports app for nine years in a row.

Some reasons for that popularity are its ability to stream baseball games and check player statistics in real time. For example, from my home in Arizona, I can stream games in Chicago, New York, and Los Angeles. The app's success has led to the development of complementary products and services for other sports such as golf and hockey.

Applying *Moneyball*-type thinking to software projects might help software teams find hidden value and operate more efficiently and effectively. Two sabermetrics concepts come to mind: quantifying a player's ability to score runs (using the *runs created* metric) and a player's ability to help a team win (using the *win probability difference* metric).

Runs Created

Bill James invented the runs-created metric to estimate the number of runs a hitter contributes to the team. To explain why this metric is essential, James used the example of Willie McCovey, a first baseman for the San Francisco Giants in the 1960s and 1970s who was inducted into the US National Baseball Hall of Fame, an honor reserved for the top 1 percent of players:

> *With regard to an offensive player, the first key question is how many runs have resulted from what he has done with the bat and on the basepaths. Willie McCovey hit .270 in his career, with 353 doubles, 46 triples, 521 home runs and 1,345 walks—but his job was not to hit doubles, nor to hit singles, nor to hit triples, nor to draw walks or even hit home runs, but rather to put runs on the scoreboard. How many runs resulted from all of these things?*[2]

James was arguing that these numbers don't tell the entire story of McCovey's career. Rather than focusing on individual metrics, James suggested that the most important metric should be how many runs resulted from McCovey's contributions. This shifts the focus from individual outcomes to team outcomes, importantly so because runs help a team win.

The conceptual framework for runs created (RC) is

$$RC = (A \times B)/C,$$

where A is the on-base factor (how many times the batter got on base), B is the advancement factor (a weighted sum of the number of bases a batter gained with his or her hits), and C is the opportunity factor (how many times a batter had the opportunity to hit).

The analog in software development is that an individual programmer's contribution could potentially be measured in terms of thousands of software lines of code created (KSLOCC). However, I suggest measuring the project team's productivity instead of individual programmer productivity, using this formula:

$$KSLOCC = (D \times E)/F,$$

where D is the KSLOC created by the team, E is the complexity weights for more difficult KSLOC, and F is the team's effort in person months.

This article originally appeared in IEEE Software, *vol. 34, no. 5, 2017.*

In other words, instead of measuring an individual programmer's productivity by using the standard KSLOC-per-person-month metric[3] at the individual-programmer level, I propose two modifications:

- Measure the KSLOC generated by the project team rather than the individual.
- Weight the software by complexity to account for more difficult features or modules.

These measures would work only when measuring productivity at the team level makes sense and when a clear definition of a team exists.

The Win Probability Difference

As I mentioned before, a team's likelihood of winning a game can be quantified in terms of probabilities throughout the game. This makes a sport more interesting for fans who might want to understand how certain plays affect the game's potential outcome. It's also exciting to see when a team with a very low probability of winning suddenly comes back and steals the win from another team.

Multiple factors drive a team's likelihood of winning a baseball game. One factor is whether the team is playing in its home stadium. Another factor is the sequence of events of the game itself. If the score is 5 to 0 at the game's early stages (with 6 of 27 outs recorded for the team that's ahead), the probability of a win for the team that's ahead will be lower than if the score was the same near the game's end (with 26 of 27 outs recorded for that team).

The events that lead to the offensive production of runs or defensive production of outs can be attributed to individual player contributions. In line with the earlier discussion about the need to emphasize team behavior over individual behavior, the win probability difference emphasizes how much a player helps his team on offense and defense.[4]

A recent example is the 29 May 2017 game between the Houston Astros and Minnesota Twins. As mentioned before, both teams began with $p(\text{win}) = 0.5$. By the fifth inning, the Twins were ahead 7 to 2, with $p(\text{win}) = 0.95$. The game remained in favor of the Twins until the Astros scored in the 8th inning, shifting the Twins' $p(\text{win})$ from 0.76 to 0.27 with a single play: a double by Josh Reddick. The difference that double made in terms of $p(\text{win})$ certainly had the most impact, despite accounting for only two of the team's 16 total runs. (For a graph of how the *win expectancy*, which is akin to $p(\text{win})$, changed throughout the game, see www.fangraphs.com/livewins.aspx?date=2017-05-29&team=Twins&dh=0&season=2017.)

The analog to software also pertains to the probability of a successful outcome, which might be cost, schedule, or performance based. A project team member might accomplish a certain milestone, make a technological breakthrough, achieve customer approval, or perform a test that could increase the project's likelihood of success. As with most project schedule estimates, there are optimistic expectations that the project will be completed on time. If these estimates were updated at each significant event, the team would know its likelihood of success.

Of course, an important difference between baseball and software is the role of external factors that influence the outcomes. In baseball, most outcomes are decided by skill or luck. Some are influenced by external factors such as weather or crowd noise. In software, external factors such as personnel turnover, financial crises, and customer delays might play a much more significant role in the project's success.

As with the introduction of any new metric, there are unintended consequences. Measuring certain things might lead to a change in behavior that's desirable in the short term but undesirable in the long term. My goal here has been to provide a different view of how to measure software projects by borrowing from the playbook professional baseball teams use to measure and evaluate their players. If this helps spark ideas and dialogue, my main goal has been met.

Just for fun, because I'm a fan of baseball analytics, I predict that the Houston Astros will win the 2017 World Series because they're one of the most data-driven teams in professional baseball, which will prove to be a differentiator in the long season.

References

1. M. Lewis, *Moneyball: The Art of Winning an Unfair Game*, W.W. Norton & Co., 2004.
2. B. James, *The Bill James Historical Baseball Abstract*, Villard, 1985.
3. B.W. Boehm et al., *Software Cost Estimation with Cocomo II*, Prentice Hall, 2000.
4. W.L. Winston, *Mathletics: How Gamblers, Managers, and Sports Enthusiasts Use Mathematics in Baseball, Basketball, and Football*, Princeton Univ. Press, 2009.

RICARDO VALERDI is an associate professor of systems and industrial engineering at the University of Arizona and a consultant to various Major League Baseball teams. Contact him at rvalerdi@arizona.edu.

Software Engineering

Rick Kazman, University of Hawaii

As the importance of software in our world increases, so do the duties, skills, and knowledge required of software engineers.

The term "software engineering" (SE) can be traced back to the title of a 1968 NATO conference. Industry had come to recognize that to create cost-effective solutions to practical problems, scientific knowledge had to be applied to software—that is, software needed to be engineered and not merely crafted. There was little understanding then of how to achieve this goal, though progress was made over the next 20 years. "Software engineering is not yet a true engineering discipline," Mary Shaw wrote in 1990, "but it has the potential to become one."[1]

It's reasonable to ask whether, in 2017, we've finally achieved the aspirations of that NATO conference nearly 50 years ago. There are clear signs we're rapidly moving in that direction. We have codified bodies of knowledge—such as the *Guide to the Software Engineering Body of Knowledge* (SWEBOK),[2] now in its third edition—and curriculum guidelines for undergraduate SE programs.[3] There are also professional bodies that license software engineers. IEEE and the National Council of Examiners for Engineering and Surveying (NCEES), for example, offer professional licenses in both computer engineering and SE.[4] In addition, the number of accredited SE programs is rapidly increasing worldwide.

Arguably, software is changing more quickly and dealing with more complex problems than any other engineering discipline. A car in 1980 contained about 50,000 LOC; today's cars contain tens of millions of LOC, and high-end vehicles contain hundreds of millions of them. A typical Linux distribution is also hundreds of millions of LOC. Clearly, we're doing something right in managing this enormous complexity. Decades of work on software abstraction[5] and patterns[6] have helped us create—and gain intellectual control over—systems of ever-increasing complexity. But SE needs to change to meet the challenges of the future. Software is everywhere in the infrastructure that surrounds us, and it affects all of us. For this reason, new dimensions of SE are gaining prominence.

A bridge or a building is typically built to last a century or more, with periodic maintenance, but software changes rapidly—in some cases daily. We've become accustomed to the flood of releases of the software that runs our lives: OSs, desktop applications, mobile apps, and utility software (such as virus scanners). In this way, SE differs from most traditional engineering disciplines: software engineers must deal with the consequences of constantly changing requirements and environments. For this reason, release engineering, continuous delivery, and DevOps have become core competencies that software engineers need to master. For example, Amazon is

 Published by the IEEE Computer Society

reputed to deploy code every 11.7 seconds, and Etsy does over 50 deployments per day;[7] Facebook updates its code at least twice per day.[8]

Furthermore, as software increasingly runs more of our world, including the emerging Internet of Things (IoT), two new areas of SE are gaining importance: green SE and social SE. The environmental consequences of software are rapidly growing; for example, datacenters now account for the same amount of greenhouse gases as global aviation.[9] And with our lives centered on smartphones, power consumption and battery life are among the quality attributes engineers must worry about.[10]

As software grows in importance and projects grow in size, software engineers need to be concerned with systems' technical qualities. A number of key technological developments, such as cloud computing, have made the deployment of ever-larger systems feasible. These systems are part of the fabric of our society—they run our power grids, our phones, the Internet, our businesses, and our government. This trend is set to grow dramatically as the IoT expands. To keep all of this continually running, our systems are becoming self-monitoring, adaptive, and self-healing. The challenge of being "always on" also highlights the importance of cybersecurity: as our world increasingly depends on software, the risks of software errors, flaws, or hacks increase correspondingly

Finally, software engineers must be aware of the sociotechnical ecosystems in which those ever-larger systems are built, maintained, and used. Software is increasingly open source and crowdsourced. Thus, software engineers need to be not just technical leaders—although clearly that's a necessary condition—but also community shepherds.[11] In addition to technical mastery of architecture, implementation, tools, and technologies, engineers need to acquire soft skills and be able to guide, persuade, negotiate, and work with others in (often global) interdisciplinary teams.

As the importance of software in our world increases, so do the duties, skills, and knowledge required of software engineers. The challenges are there and we, as a community, must rise to meet them.

REFERENCES

1. M. Shaw, "Prospects for an Engineering Discipline of Software," *IEEE Software*, vol. 7, no. 6, 1990, pp. 15–24.
2. P. Bourque and R.E. Fairley, eds., *Guide to the Software Engineering Body of Knowledge Version 3.0*, IEEE Computer Society, 2014.
3. "Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering," IEEE Computer Society and Assoc. for Computing Machinery, 23 Feb. 2015; www.acm.org/education/se2014.pdf.
4. A. Collins, "Professional Licensure for Computer Engineers and Software Engineers," Insight IEEE USA, 14 Apr. 2015; insight.ieeeusa.org/insight/content/careers/97473.
5. "On the Criteria to Be Used in Decomposing Systems into Modules," *Comm. ACM*, vol. 15, no. 12, 1972, pp. 1053–1058.
6. F. Buschmann et al., *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*, Wiley, 1996.
7. C. Null, "10 Companies Killing It at DevOps," TechBeacon; techbeacon.com/10-companies-killing-it-devops.
8. E. Protalinski, "Facebook Now Updates Its Code Twice Every Day," CNET, 3 Aug. 2012; www.cnet.com/news/facebook-now-updates-its-code-twice-every-day.
9. A. Vaughan, "How Viral Cat Videos Are Warming the Planet," *The Guardian*, 25 Sept. 2015; www.theguardian.com/environment/2015/sep/25/server-data-centre-emissions-air-travel-web-google-facebook-greenhouse-gas.
10. L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 3rd ed., Addison-Wesley, 2012.
11. D.A. Tamburri, R. Kazman, and H. Fahimi, "The Architect's Role in Community Shepherding," *IEEE Software*, vol. 33, no. 6, 2016, pp. 70–79.

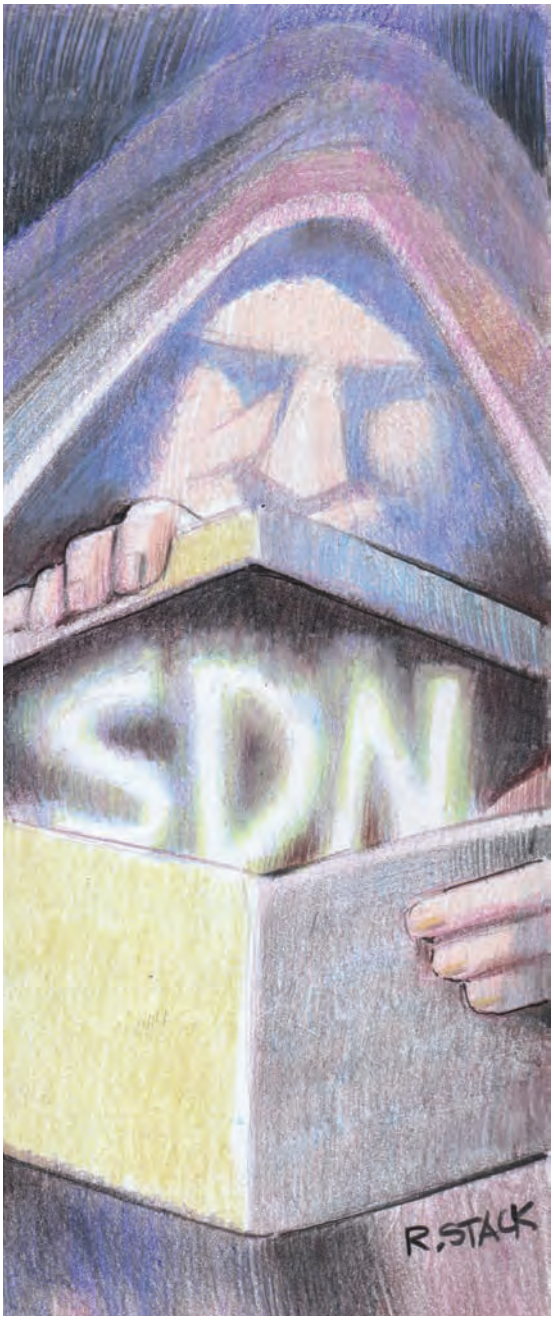
RICK KAZMAN is a professor of information technology management at the University of Hawaii and chair of the IEEE Technical Council on Software Engineering. Contact him at kazman@hawaii.edu.

This article originally appeared in Computer, *vol. 50, no. 7, 2017.*

Editors: William Enck, whenck@ncsu.edu | Thorsten Holz, thorsten.holz@rub.de | Angelos Stavrou, astavrou@gmu.edu

Security Challenges and Opportunities of Software-Defined Networking

Marc C. Dacier | Qatar Computing Research Institute
Hartmut König and Radoslaw Cwalinski | Brandenburg University of Technology Cottbus
Frank Kargl | University of Ulm
Sven Dietrich | City University of New York



Since the beginning of the decade, software-defined networking (SDN) has attracted much attention from both industry and academia, and this trend continues today. In 2016, the market research firm International Data Corporation (IDC) predicted that the market for SDN network applications would reach US$3.5 billion by 2020.[1] Especially in industry, the vision of "programming computer networks" has electrified many IT managers and decision makers. Consequently, expectations are high regarding SDN's promise. Leading IT companies such as Nokia, Cisco, Dell, HP, Juniper, IBM, and VMware have developed their own SDN strategies. Major switch vendors as well as many promising start-ups offer SDN-enabled switches.

Background

In essence, SDN provides a way to virtualize network infrastructure—to simplify it and to configure and manage the network centrally. It separates the control plane in routers and switches, which decide where packets are sent, from the data plane, which actually forwards the traffic to its destination. SDN allows control over network flows from a centralized control application running on a server or virtual machine. This controller creates rules for how network traffic is handled and routed in the network. Rules are then installed in network forwarding devices. In a sense, the routers and switches become "slaves" of this application-driven controller.

SDN-enabled networks are capable of supporting user requirements from various business applications (service-level agreements, quality of service, policy management, and so on). Most SDN approaches rely on the widely used OpenFlow protocol to provide communication between controllers and networking equipment.[2] OpenFlow is a vendor-independent standard and thus allows for interoperability between heterogeneous devices. Besides centrally defined routing policies, another key advantage of SDN is that it allows routing choices to be defined at a much finer granularity level, that is, per flow rather than at the usual IP-prefix level. For instance, OpenFlow 1.5 supports 44 different types of header fields against which to match a packet in order to choose the flow it belongs to and, thus, determine the route it should follow.

Security Aspects

Despite the enthusiasm surrounding SDN, its security-related aspects have only more recently been considered.[3,4] Opinions differ widely. Some believe that the security problems introduced by SDN are manageable—that SDN can even bring security benefits; others think that Pandora's box has been opened because SDN-enabled networks will be so complex that they will become extremely difficult, if not impossible, to properly secure.

Published by the IEEE Computer Society

The SDN paradigm has definitely accelerated the discussion about new, more efficient methods for controlling and managing computer networks. SDN is also a means to implement new security mechanisms, introducing them into systems in ways that weren't previously possible—which is good. Detecting attacks might become easier and more reliable, but SDN also increases the attack surface, and its standards notoriously lack appropriate security mechanisms, such as for authorization[5]—which is bad. In particular, having regular applications introduce "network apps" that interact with the controller to modify network behavior based on application demands could be a complexity nightmare in terms of the required authentication and authorization schemes. Thus, whether the SDN paradigm's good or bad aspects will prevail and where the final balance will be remain open questions.

These two contrary facets of SDN security were the key ingredients of the lively and very fruitful Dagstuhl research seminar that we hosted in September 2016. (See the sidebar for background information on Dagstuhl seminars.) The seminar brought together experts from industry and academia, the security and networking communities, and the pro- and anti-SDN camps alike.[5] The objectives of the seminar were to

- discuss the security challenges of SDN,
- debate strategies to monitor and protect SDN-enabled networks, and
- propose methods and strategies to leverage SDN's flexibility for designing new security mechanisms.

The seminar began with a discussion of SDN's good and bad aspects from a security viewpoint. The participants then identified open issues and formulated research directions toward achieving more secure SDN. Generally, participants agreed that although SDN provides new possibilities to better secure networks, it also poses several serious security problems that require further research. Without finding adequate solutions for the latter, SDN can't be successfully and securely applied on a broad scale.

Dagstuhl Seminars

Dagstuhl seminars (www.dagstuhl.de/en/program/dagstuhl-seminars) have a worldwide reputation in the computer science community and beyond as a premier place for in-depth scientific exchanges and discussions. The seminars, held in the scenic countryside of Saarland in southwestern Germany, typically last one week and are initiated by up to four organizers, each representing the different invited communities. On the organizers' behalf, The Schloss Dagstuhl–Leibniz Center for Informatics invites 35 to 45 researchers of international standing from academia and industry, including many promising young researchers. Dagstuhl seminars typically don't have a set program; instead, the pace and program are guided by topics and presentations as they evolve through discussions.

The Good and the Bad

Drawing on the discussions at our Dagstuhl seminar, we summarize here the participants' main exchanges and conclusions.[6]

Centralization in SDN

By design, SDN centralizes many networking aspects that have traditionally been decentralized. For example, SDN-driven networks might offer a centralized location to manage the data plane. New algorithms assume a centralized data model, which wasn't possible in traditional networking. This is a radical change. SDN increases network complexity, and the plentitude of intertwining algorithms might emit contradicting security policies. However, you could also argue that centralization allows you to resolve such inconsistencies.

The good is that reacting to and removing such policy inconsistencies are much easier in a centralized manner. This has positive implications for many policy types, including centralized routing algorithms, firewalls, and network-monitoring methodologies.

The bad relates to the well-known issue of single point of failure, which downgrades a distributed system's resilience. It's debatable whether traditional networks don't already offer single points of failure, but SDN adds some additional centralization points that might be exploited by an internal attacker who suddenly has a central place to monitor and manipulate the network, or by an external adversary who needs to compromise fewer vulnerable SDN components to gain full network control. Further thought is required regarding how SDN can be protected against such attacks. In addition, SDN's centralized decision engine adds a new type of denial-of-service (DoS) vector. Indeed, an attacker could overload the controller with unknown flows that require constant decision-making. On the other hand, SDN's centralization allows more effective management of existing DoS attack types, because it has a global view of the network topology and can correlate this information with the traffic analysis to more reliably detect attacks.

The centralization imposed by SDN creates new challenges, but the benefits appear to be predominant.

However, it's important to address the open research questions in this regard to ensure centralized SDN systems' security and resiliency.

What Benefits More—The Attack Surface or Opportunities for Defense?

SDN's good relates to the advantages it provides to defenders and the limitations it poses to attackers. With its global view of the network, open hardware interfaces, and a centralized control, SDN's centralized architecture supports the defenders and allows them to create tailored security solutions, such as for anomaly detection in wireless networks.[5] The bad is that a centralized architecture, lack of defender expertise, and still immature technology benefit the attackers. Attacks against SDN controllers and the introduction of malicious controller apps are probably the most severe threats to SDN.[3,7]

Flexibility and Adaptability for Attackers and Defenders

The good—SDN's real added value to security—is its ability to interact with switches and routers by means of APIs. These APIs can be leveraged for many security-related tasks independent of complete adoption of the SDN paradigm. For defenders, it gets easier to statically or dynamically isolate networks, refine client authentication and authorization, enable active response (blocking, restricting), gain network overview for creating awareness on the current security situation, adaptively monitor the network, and improve resilience when under attack. For attackers, attack-related activities such as network reconnaissance, analysis of properly separated network environments, man-in-the-middle attacks using spoofing, and system takedown get harder.

The bad—SDN's potential problems—mainly relates to increased complexity, for example, having to configure SDN-capable switches from various sources and by different users. This allows attackers to control the operations in arbitrary ways, confuse or blind the defenders, and create inconsistencies. They can gain a global and more fine-grained view of the network from a single location. Furthermore, they can exploit the additional complexity caused by flexibility (for instance, manipulation attacks on the switch and controller side). Dynamic configurations make it more difficult for defenders to tell whether the current or past configuration is intended and correct. The more user-friendly tools get, the less humans can intervene manually and develop a deep understanding of the underlying technology and protocols. In addition, flexibility makes it hard to define meaningful SDN network policies, such as which flows are affected by a specific network application and modified in a specific way. The flexibility SDNs provide might amplify conflicts between networking objectives and security demands.

Software-defined networking offers advantages for securing networks but also raises questions about new vulnerabilities.

Is SDN More Complex, or Is It Simpler?

SDN promises—and this is the good—reduced complexity by splitting networks into a dedicated data plane and a logically centralized control plane. For concepts such as routing, the software approach in SDN seems much simpler than the distributed approach in classical networks. This narrative is countered by two aspects hidden in the simplistic SDN model regarding the controller as a single entity rather than a distributed system: the need for scalability and operational requirements, for example, concerning fault tolerance. Both strongly call for the use of a distributed approach. In addition, implementing the control plane completely in software raises issues about its algorithmic complexity. This is due to additional requirements that weren't imposed on classical networks but are now thinkable in SDN. Although this is a unique selling point in terms of possible features, it raises serious security concerns because it opposes simplicity, which is a key design principle in building secure systems. Separating the data and the control plane creates different views and, with emphasis on their consistency, makes creating a holistic security solution tough. SDN introduces numerous challenges regarding system complexity and simplicity. It has the potential to be simple—but making it simple is quite complex. The decomposition of components is easy, but their secure reassembly remains challenging. Therefore, a self-limitation regarding the necessity of features must be considered to allow the simple and secure design, implementation, and operation of SDNs.

Research Directions

The full Dagstuhl seminar report offers more insight into the pros and cons of SDNs as related to security, and summarizes the various working groups' discussions.[6] The outcomes of these discussions led to several research directions that have been refined along four axes. We briefly review each here.

Automated Derivation of Secure SDN Configurations

Automatically deriving secure SDN configurations is one important research goal. This requires extending state-of-the-art methods by providing additional information elements for the full range of network components representing all states of SDN network elements. Improvements are also required in the methods available to assess SDN networks' security, which range from penetration testing to formal analysis tools, such as policy checkers.

Secure Operations in SDN Environments

SDN is unlikely to completely replace existing, non-SDN-based environments or to be totally disruptive rather than incremental. Mixed operations and stepwise introduction of SDN will lead to issues that will require further research to be fully understood and mitigated. SDN's human and organizational dimensions must be carefully scrutinized to identify who's in charge of what and why, and to proactively identify possible conflicts that could make the system insecure. In particular, this was argued for SDN applications responding to possibly conflicting requirements from different applications or organizations. Last but not least, there were concerns that SDN environments would rejuvenate old threats such as covert channels and expand the attack surface in unanticipated ways.

SDN-Based Security

The full report lists several typical attacks and considers how SDN can enable not only new network security mechanisms to prevent, detect, or react to such attacks but also better forensics analysis.[6] At a high level, two main capabilities were identified that security mechanisms should use to better secure SDN-based environments. First, SDN and OpenFlow allow holistic control of network devices throughout all active network components, allowing traffic to be inspected or filtered anywhere. Second, SDN offers a standardized interface for interacting with the network, allowing the implementation of cross-platform security mechanisms. These areas have already received some attention but deserve further research.

Secure Architecture for SDN

Last but not least, participants discussed applicable architectural patterns and best practices that should be made available to a broad audience to improve security. Whereas they agreed that such architecture is needed, they also felt that a solution to this problem would require deeper, longer research. In particular, they proposed several key questions to be answered by such architecture. Networking apps' concept and role in such an architecture were general concerns.

SDN is here to stay, but its precise definition keeps changing. It offers numerous advantages for securing networks, but it also raises questions related to new vulnerabilities and an increased attack surface—questions that must be faced honestly. Simple SDN solutions foster SDN security, but keeping SDN simple is complex! Securing SDN networks will require secure SDN network applications and composition thereof. This requires further research on not only network security but also secure software engineering. Without a clear SDN security research road map, SDN's benefits might be quickly overcome by its security issues, and we'd be left with a difficult choice: open Pandora's box by deploying insecure SDN networks, or stop benefiting from a promising networking communication paradigm.

We hope that the insights we've shared provide SDN researchers with valuable directions to improve their SDN applications' security and contribute to the construction of secure software-defined networks. ■

Acknowledgments

We thank the Dagstuhl Seminar 16361 participants, whose contributions provided the basis for this article.

References

1. B. Casemore and B. Mehra, *SDN Market to Gain Enterprise Headway, Driven by 3rd Platform and Cloud*, tech. report US40628315, IDC, Nov. 2015; www.idc.com/getdoc.jsp?containerId=US40628315.
2. N. McKeown et al., "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Computer Communication Rev.*, vol. 38, no. 2, 2008, pp. 69–74.
3. S. Scott-Hayward, S. Natarajan, and S. Sezer, "A Survey of Security in Software Defined Networks," *IEEE Communication Surveys & Tutorials*, vol. 18, no. 1, 2016, pp. 623–654.
4. C. Röpke and T. Holz, "Retaining Control over SDN Network Services," *Proc. Int'l Conf. and Workshops Networked Systems* (NetSys 15), 2015; doi.org/10.1109/NetSys.2015.7089082.
5. R. Cwalinski and H. König, "RADIator—An Approach for Controllable Wireless Networks," *Proc. NetSoft Conf. and Workshops* (NetSoft 16), 2016; doi.org/10.1109/NETSOFT.2016.7502421.
6. M. Dacier et al., "Network Attack Detection and Defense: Security Challenges and Opportunities of Software-Defined Networking," *Dagstuhl Reports*, vol. 6, no. 9, 2016, pp. 1–28; drops.dagstuhl.de/opus/volltexte/2017/6912/pdf/dagrep_v006_i009_p001_s16361.pdf.
7. D. Kreutz, F. Ramos, and P. Verissimo, "Towards Secure and Dependable Software-Defined Networks," *Proc. ACM SIGCOMM Workshop Hot Topics in Software-Defined Networking* (HotSDN 13), 2013, pp. 55–60.

Marc C. Dacier is a research director at Qatar Computing Research Institute. Contact him at marc.c.dacier@gmail.com.

Hartmut König is a professor at Brandenburg University of Technology Cottbus. Contact him at hartmut.koenig@b-tu.de.

Radoslaw Cwalinski is a PhD student at Brandenburg University of Technology Cottbus. Contact him at radoslaw.cwalinski@b-tu.de.

Frank Kargl is a professor at the University of Ulm. Contact him at frank.kargl@uni-ulm.de.

Sven Dietrich is an associate professor at the City University of New York. Contact him at spock@ieee.org.

This article originally appeared in
IEEE Security & Privacy, *vol. 15, no. 2, 2017.*

IEEE COMPUTER SOCIETY: Be at the Center of It All

IEEE Computer Society membership puts you at the heart of the technology profession—and helps you grow with it.

Here are 10 reasons why you need to belong.

IEEE Computer Society—keeping you ahead of the game. Get involved today.

www.computer.org/membership

IEEE computer society

Editor in Chief: **Diomidis Spinellis**
Athens University of Economics
and Business, dds@computer.org

Software Reliability Redux

Diomidis Spinellis

SOFTWARE-INTENSIVE SYSTEMS with high reliability requirements typically are implemented through heroic (and expensive) engineering efforts. Control systems in planes, nuclear reactors, trains, pacemakers, and spaceships are developed by highly trained personnel through strictly managed software development processes with a dose of formal methods. This approach has worked admirably up to now, but its strains are beginning to show.

We're Not in Kansas Anymore

Start with ubiquity and cost. With "software eating the world," the requirement for high reliability is no longer restricted to a few specialized and proven domains. Instead, ever more functions whose failure can hurt humans and damage property are cropping up in new areas. Critical software appears in applications ranging from hobbyist drones and Wi-Fi routers to lithium-ion battery charging circuits and personal health monitors, to automated trading and door locks. Frighteningly, the software development budget for some application areas might be too low to cover fancy reliability engineering. So, the organizations that develop the software might lack the people, processes, and tools to deliver the required reliability.

Then there's the risk from end-user programming. Software applications increasingly offer users the ability to configure and program them. This can be helpful when we use a spreadsheet to automate submission of our travel expenses or use a content management system to simplify editing our school's website. However, letting untrained users program in critical application areas could be like letting a drunk pilot fly a jumbo jet.

This state of affairs often develops gradually, in ways that are difficult to manage. An enthusiastic amateur programmer realizes he or she can use a small Visual Basic or Python script to easily automate a peripheral but tedious process. Over the years, the process becomes more important to the amateur programmer's organization, and the script grows multiple tentacles as it gets connected to other services. Then, a user mistakenly enters a negative price or the script runs on 29 February, and multiple

IEEE Software Mission Statement

To be the best source of reliable, useful, peer-reviewed information for leading software practitioners—the developers and managers who want to keep up with rapid technology change.

 Published by the IEEE Computer Society 2469-7087/17/$33.00 © 2017 IEEE

services fail catastrophically because the script was never properly tested.

Critical software with high reliability requirements is also growing bigger and more complex. This happens because, spurred by advancements in other application areas and increased hardware capabilities, we demand more from it. For example, we expect a car's console to be at least as friendly as our smartphone, not realizing that a software crash on our phone is an inconvenience, whereas a car crash can be a tragedy.

In addition, managing the development of critical software becomes more difficult because the way we build software is changing, with third-party components providing much of an application's required functionality. The Apollo program's spacecraft software ran on bare metal, and each part of it could be carefully verified. In contrast, a modern critical-application software stack might include an OS kernel with many millions of lines; third-party device drivers and firmware in binary form; large middleware components; and open source libraries handling data compression, HTTP communication, or cryptography developed by thousands of volunteers.

As if handling the size and complexity wasn't challenging enough, many software applications requiring high reliability comprise a multitude of interconnected systems. Parts of an application might run in an embedded device; other parts might run on a cloud provider's servers; and yet other elements might depend on queuing, geolocation, image recognition, messaging, or database functionality provided by third parties as a service. These complex systems' failure modes are difficult to predict and handle. Famously, when some of Amazon's cloud services failed a few months ago, the status indication dashboard didn't work as expected because the necessary red or green images were stored on Amazon's failed Simple Storage Service.

To top it all, critical software often must be actively maintained for decades. As Mike Milinkovich, the Eclipse Foundation's executive director, said, "The software you're writing today may have to be maintained by your great-granddaughter."[1] This has always been the case because the time span from design to the end of the corresponding hardware's life can indeed be more than half a century. What has changed is the type of required maintenance. Systems connected over the Internet require regular updates to face new threats and to handle protocol evolution. It was admirable that Microsoft had in place a build environment and an infrastructure to release a Windows XP patch for the EternalBlue vulnerability later exploited by the WannaCry ransomware. However, the organizations whose operations relied on the long-unsupported system were treading on thin ice. Also, the hardware of modern large complex systems depends on so many manufacturers that maintaining it in its original state for decades is hard. The necessary upgrades bring with them new device drivers and fresh whole OS releases—a verification nightmare for critical systems.

Somewhere, over the Rainbow, Skies Are Blue

Avoiding problems and catastrophes in the new software reliability landscape won't be easy. Consider the ubiquity of software performing critical functions and of devices whose software isn't appropriately maintained. Unfortunately, for software that's developed with opaque,

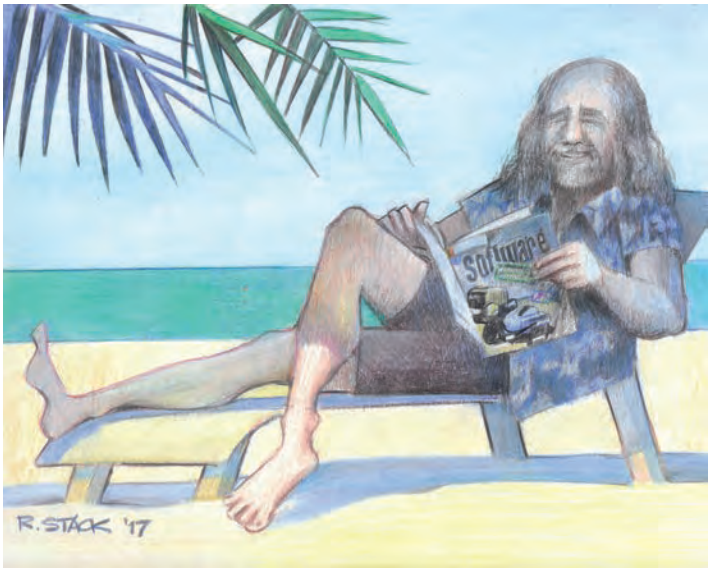
potentially slapdash, processes, part of the answer will likely have to be regulation. Currently, the cost of misbehaving software is passed to users (in the form of failures) and the environment (as devices discarded owing to faulty unmaintained software). Left on its own, the market is unlikely to solve this problem. This is because users have insufficient information regarding the software's reliability and because most software isn't marketed in time frames that allow the establishment of trustworthy brands. So, regulation that increases transparency regarding the software's reliability and makes manufacturers of critical software liable for failures and responsible for maintenance over clearly specified periods will result in better outcomes for all parties involved.

The issues associated with end-user programming will require multiple parties to do their part. Organizations must set up efficient methods to inventory and characterize their software assets and the assets' dependencies and importance. In parallel, developers of applications and frameworks that are often used for end-user programming must continue promoting the development of more reliable systems. Some avenues include increased reliance on static checking; runtime provisions for handling and recovering from failures; and built-in support and gentle encouragement for good software development processes such as modularization, unit testing, and configuration management. Given the ever-larger number of people involved in putting together algorithmic rules and systems, increased software engineering literacy among the general population will also help.

There are no easy answers to the reliability challenges arising from modern software's size and complexity. Making suppliers responsible for software maintenance and failures should result in the availability of more trustworthy components. As a bonus, in such an environment, we'll be more likely to see a business case for maintaining critical open source libraries and systems. Thankfully, systems software, which faces less pressure to evolve to changing requirements than applications do, becomes more reliable as it matures. So, designers should prefer us-

THANK YOU, GRADY!

Grady Booch published his first *IEEE Software* article in 1994[1] and graced our magazine with his On Architecture and On Computing columns from 2007 until last year. I've learned a lot from his thoughtful, original, and reflective writing, a feeling I'm sure all *IEEE Software* readers share. So, please join me in thanking Grady for his long, gallant service to our magazine and community.



Reference

1. G. Booch, "Coming of Age in an Object-Oriented World," *IEEE Software*, vol. 11, no. 6, 1994, pp. 33–41.

ing software components that have proved their mettle over the temptation to adopt whatever technology is in fashion each year.

Addressing reliability concerns is even more difficult with complex systems. Few organizations and groups have experience developing and running complex, large, reliable systems. Even those organizations with that experience have occasionally contended with spectacular failures.

Thus, the first lesson is to isolate the most critical functionality in stand-alone units rather than implement it as part of a complex system. We can also try to learn from experienced organizations. Commendably, some are publishing their practices[2] and failure postmortems. These lessons need to be generalized into scientific theory and make their way into university curricula. In the longer term, we can copy nature and build complex systems by combining multiple, diverse, interchangeable components with independent failure modes.

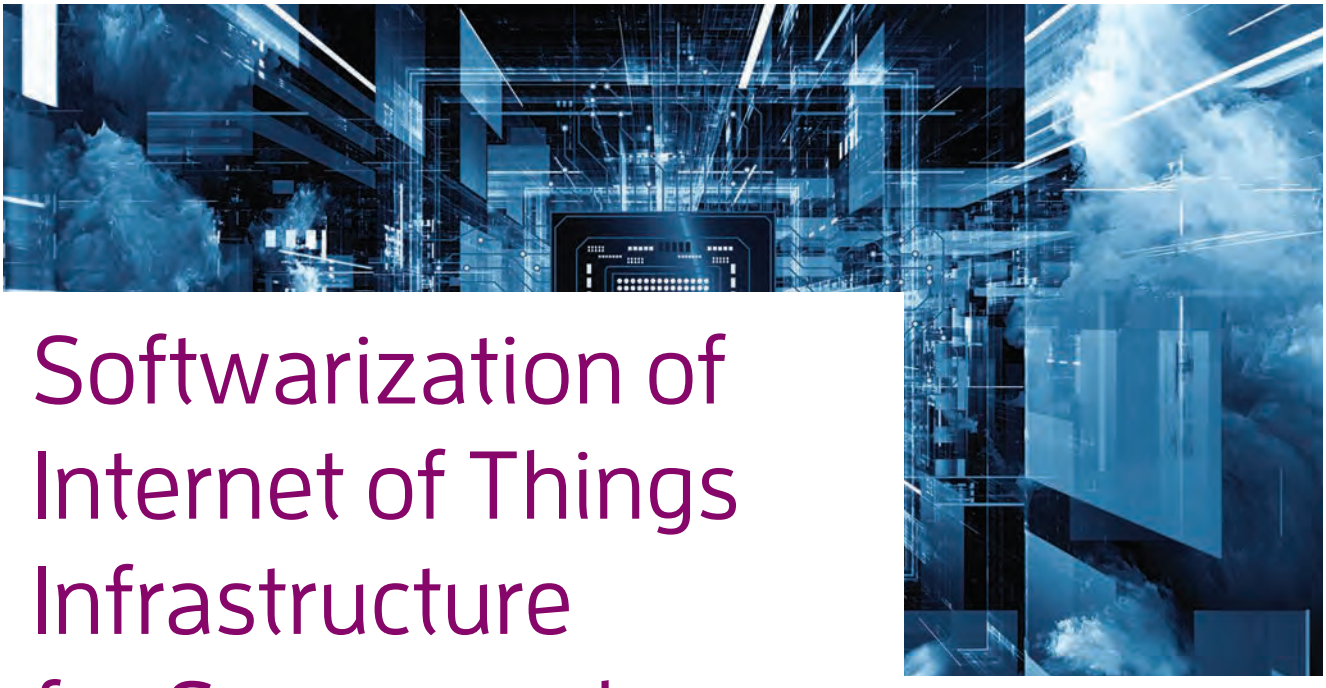
Some candidate solutions crosscut all problem areas. Innovations that reduce the cost and time to develop reliable software would help a lot, but we can't bank on them. Improved, probably longer, education with increased emphasis on software reliability can be a requirement for people developing critical software. As professionals, we should also assume more responsibility for the software we develop. Professional societies can do their part here by standardizing and promoting the state of the art. An admirable step in this direction is the IEEE Computer Society's *Guide to the Software Engineering Body of Knowledge* (available at www.computer.org/web/swebok/v3).

Throughout its 50-year history, software engineering has evolved splendidly through numerous crises. Modern software reliability challenges can also be solved by applying the two simple elements used in all past calamities: the courage to face the problem and the brain to solve it.

References

1. M. Milinkovich, "Open Collaboration: The Eclipse Way," keynote address at 2017 Int'l Conf. Software Eng. (ICSE 17), 2017.
2. B. Beyer et al., *Site Reliability Engineering: How Google Runs Production Systems*, O'Reilly Media, 2016.

This article originally appeared in IEEE Software, *vol. 34, no. 4, 2017.*



Softwarization of Internet of Things Infrastructure for Secure and Smart Healthcare

Mohammad A. Salahuddin, University of Waterloo
Ala Al-Fuqaha, Western Michigan University
Mohsen Guizani, University of Idaho
Khaled Shuaib and Farag Sallabi, United Arab Emirates University

The authors propose an agile, softwarized infrastructure for the flexible, cost-effective, secure, and privacy-preserving deployment of Internet of Things systems for smart healthcare applications and services.

Smart healthcare will be the most dominant Internet of Things (IoT) application.[1] It will let healthcare providers leverage cloud and fog computing to optimize services while minimizing operating and capital expenditures.

The smart applications and services that the industry will use will require the collection, aggregation, and analysis of raw sensor data.[1] The many ambient and embedded devices in our environment generate large amounts of data (including text, audio, and video), which will, in various cases, require batch, pseudo-real, or real-time processing. The challenge here lies in aggregating heterogeneous data from different types of sources.

To meet this challenge, we have designed an agile, softwarized infrastructure that embraces cloud and fog computing, blockchain, Tor, and message brokers for flexible, cost-effective, secure, and private IoT deployment for smart-healthcare applications and services.

We propose a system architecture for our infrastructure, a novel platform with machine-to-machine (M2M) messaging, rule-based beacons for seamless data management, and the use of data fusion and decision fusion to facilitate smart-healthcare applications and services.

 Published by the IEEE Computer Society 2469-7087/17/$33.00

EDITOR **SAN MURUGESAN**
BRITE Professional Services; san@computer.org

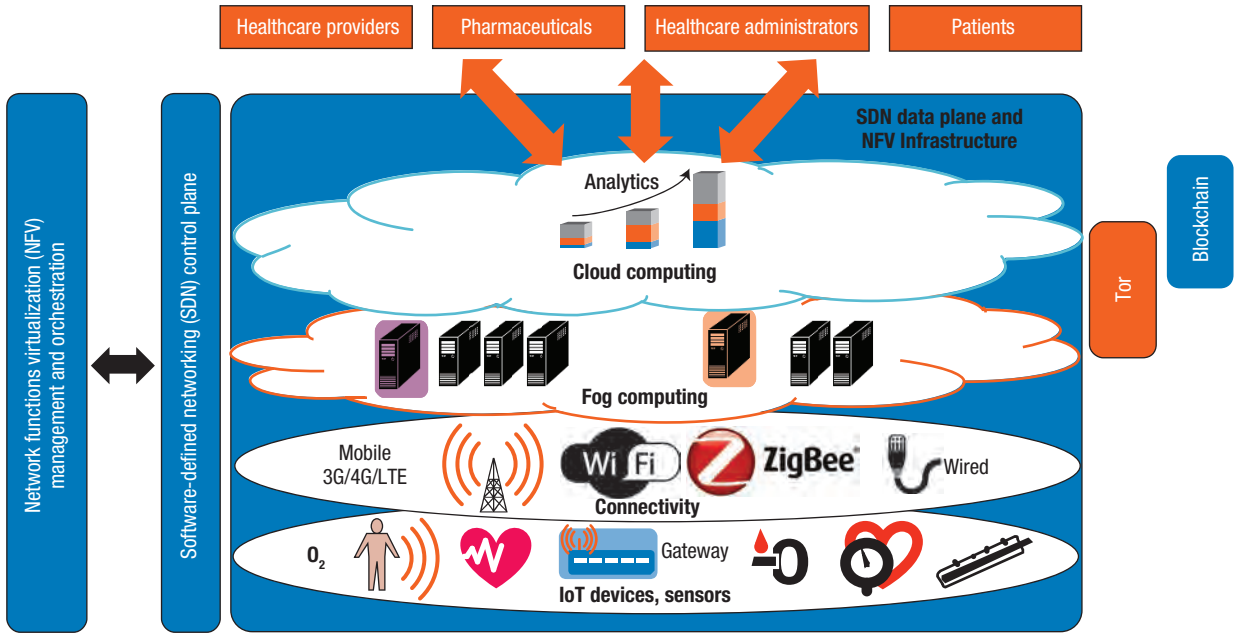


Figure 1. Architecture for a softwarized Internet of Things (IoT) system for smart-healthcare applications and services.

SYSTEM ARCHITECTURE

Figure 1 illustrates the interplay of our system's various IoT elements and its networking and computing technologies. The elements include various sizes and types of smart sensors that monitor patients' health-related parameters and that also process and record raw sensor data. Transceivers on the sensors communicate with base stations via a wireless interface. The more powerful base stations act as data aggregators, sink nodes, or gateways to the cloud.[2] The IoT gateways work with the different types of devices and network protocols involved and thereby enable general connectivity.[3]

Softwarization

Usually, sensor networks are application-specific and aren't dynamically configurable. However, software-defined networking (SDN) can economically improve sensor networks' agility and flexibility. SDN decouples a network's control and data planes, and allows the dynamic and flexible configuration and management of data-forwarding rules. This improves interoperability among communication protocols[4] and reduces the cost of network deployment, configuration, and management by letting users easily make commercial off-the-shelf (COTS) hardware SDN-compliant.[5]

The softwarized infrastructure can also program COTS hardware to perform network functions and even deliver end-to-end services[6] via network-functions virtualization (NFV). The softwarized infrastructure connects the virtual network functions (VNFs) to compose a service, while the SDN controller helps steer traffic between virtual and physical network functions and applications. A software NFV manager and orchestrator creates, configures, manages, and monitors the VNFs. The IoT gateways directly or indirectly connect to the VNFs, which promotes agility and cost-effective application and service delivery.

Figure 1 shows how the cloud-based system can optimize healthcare delivery[7] via analytics, which enables early detection and prevention of projected patient risks, identifies possible disease outbreaks or epidemics, and improves healthcare-delivery precision. The cloud augments comprehensive patient records with biometric,[8] genomic,[8] familial,[8] and social[9] data, giving healthcare providers a holistic perspective of patients' mental, physical, and social status. The system also identifies waste and resource misuse, which reduces operating and capital expenditures. And the cloud hosts and thereby increases the security, privacy, and resiliency of data, applications, and services.

Security and privacy

Mechanisms such as Tor,[10,11] in tandem with M2M protocols like MQTT

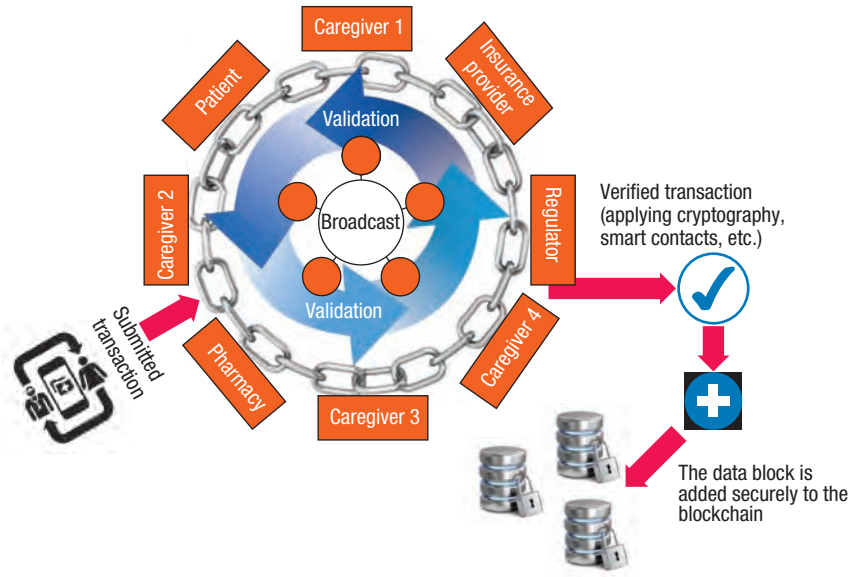


Figure 2. Using blockchain technology to secure patient records. A blockchain is a decentralized and distributed database that validates, records, timestamps, and maintains all transactions in a network of computers available only to authenticated participants. All transaction blocks are visible to doctors, patients, and other participants, who could spot any unauthorized data manipulation. The decentralized network lacks centralized points of vulnerability that hackers can exploit.

(Message Queue Telemetry Transport), preserve user and data anonymity, counter network surveillance threats, and thereby protect the privacy of patient records and other sensitive healthcare information.

Tor enables anonymity by imposing an overlay network of secure connections between nodes in the underlying network and selecting multiple random communication paths. Because Tor can introduce communication delays[12] and unpredictability, we propose employing it between fog nodes (which are at the network edge) and the cloud. Not deploying it end to end—for example, between a user and a cloud server—helps the system meet its stringent real-time requirements. To properly decide whether to accept Tor's tradeoff between anonymity and latency, users must understand the requirements of the application they're working with.

Blockchain technology promises to guarantee the security of patient records by tracking and authorizing access to confidential medical records, as Figure 2 illustrates. A blockchain is a decentralized and distributed database[13] that validates, records, timestamps, and maintains all of its transactions in a trusted peer-to-peer network of computers. The system makes transactions available only to authenticated participants via public-key cryptography. Because all transaction blocks are visible to participants—such as caregivers, hospitals, pharmacies, insurance companies, regulators, and patients—it's extremely difficult for adversaries to manipulate data or transactions without being noticed. In addition, the system's network lacks centralized points of vulnerability that hackers can exploit.

Latency

Many healthcare applications and services can't afford the latency that the cloud causes. In these cases, fog computing brings cloud-like resources and computing closer to users. The fog nodes in Figure 1 are smaller in size and resources than cloud nodes but more powerful than IoT devices and gateways. This low latency and high performance enables the system to efficiently process and aggregate localized data and reduce unnecessary traffic to the cloud.

DATA AGGREGATION

Users must aggregate healthcare data collected from our IoT sensor network for analytics, applications, and services. We illustrate this with a specific use case entailing two small sensor networks that are part of an IoT system for the monitoring of cardiology patients. The application requires the real-time monitoring and logging of patient data in the cloud for detailed analytics of healthcare quality and cost, as well as the ability to issue real-time alerts if it detects health problems.

One of the sensor networks consists of electrocardiogram (ECG) sensors, which measure the electrical signals that control the heart's expansion and contraction. The other consists of photoplethysmogram (PPG) sensors, which use light to sense the blood-flow rate. Each sensor network has a gateway. The ECG gateway is also an IoT gateway, as well as a fog node connected to a cloud-based healthcare database and server.

The monitoring application requires collaboration among the ECG and PPG sensors.

Data processing

There are two fundamental data-processing approaches: data fusion and decision fusion.

In data fusion, sensors transmit their raw data to a base station or a sensor-network gateway, which filters the data and reaches a decision about the patient. The decision could be about the status of a health parameter such as blood-oxygen level or about whether a health condition requires

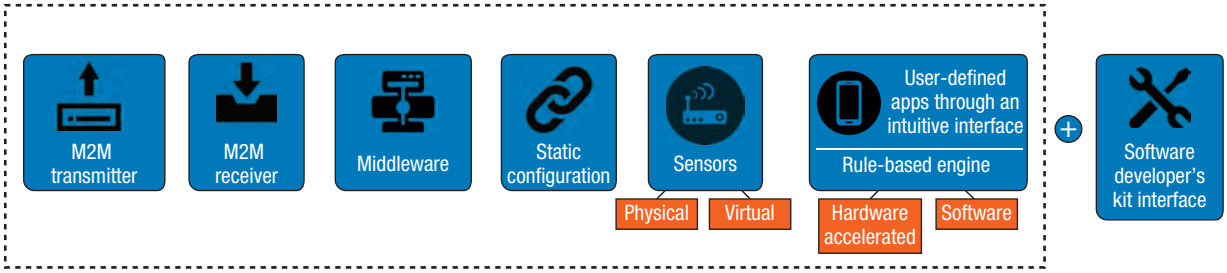


Figure 3. flexBeacon platform. flexBeacon is a deep field-reprogrammable platform with hardware and software components that deliver location- and context-aware services in resource-challenged environments via machine-to-machine communications. The system collects data from attached sensors, performs correlations, harvests analytics, and generates reports and alarms via user-defined data-flow and rule-based models.

attention. The gateway then sends the data to the IoT gateway for logging and reporting.

In decision fusion, smart sensors locally process their own raw measurements and compute the decision about what's happening with the patient's heart. The ECG sensors then transmit their data to the ECG gateway. The PPG gateway collects the blood-flow-rate data and transmits it to the ECG's IoT gateway. At the IoT gateway, the system aggregates and processes the decision about the patient's heart, and logs it in cloud-based healthcare servers for data analytics and subsequent reporting to authorized healthcare personnel's mobile devices.

There are advantages and disadvantages to using data fusion or decision fusion in IoT systems. Because data fusion transmits potentially high volumes of complex raw sensory data via radio, it consumes more bandwidth and power than decision fusion. However, decision fusion tends to be less precise because the sensors might not have highly accurate processing capabilities. Data fusion is more precise because the system performs computations on the more powerful sensor network gateways.

Agile IoT platform

Our IoT-enabled smart healthcare system uses data and decision fusion to varying degrees. We propose a platform, called flexBeacon, that utilizes a deep field-programmable gate array with hardware and software components that help personalize patient care by delivering location- and context-aware services using M2M communications, as Figure 3 illustrates.

flexBeacon offers a system that will work with different kinds of hardware so that almost any sensor or actuator can connect to it. This facilitates the sharing of telemetry and of access to remote control services. This also lets healthcare providers configure rules and data-flow models to customize healthcare monitoring and control applications.

In addition, flexBeacon allows users to define rules and logic to control the flow of data for monitoring systems and applications that control the actuation of medical devices such as insulin pumps. It also enables seamless data aggregation and analytics to streamline decision making for patients, healthcare providers, and medical-facility administrators. The M2M-based communication and the FPGA-based hardware reduce latency and improve the system's data collection, aggregation, correlation, and reporting.

Our proposed platform offers seamless data aggregation and management efficiently and without a loss of accuracy. The system also greatly reduces the cost of providing softwarized IoT for smart healthcare.

CHALLENGES

Our proposed system faces various challenges.

IoT softwarization

Softwarized IoT systems like ours will have to integrate seamlessly with 5G wireless technology, which promises ultra-low latency. This would let users interact with the system immediately via mobile devices. Researchers have evaluated the performance of existing softwarization technologies, such as SDN and NFV, with 5G systems.[14] Key challenges include the lack of a standard 5G definition, as well as questions about whether the softwarization technologies will be able to take advantage of 5G's promised multitenant and multivendor capabilities.[4]

Other concerns for softwarization technologies include managing resources, spectrum, and transmission power; achieving optimal connections between network devices, transceivers, and physical elements such as routers, fog nodes, and sensors; and providing services with different quality-of-service levels.

We will have to scrutinize further some SDN and NFV features. For example, we must define the key indicators for gauging the performance of softwarized network elements,

functions, and applications. We also must learn how to design and manage distributed controllers and network functions to ensure vertical and horizontal scalability, and how to autonomously orchestrate network functions and services across the softwarized middleware.

Security and privacy

Two key challenges are protecting data against malicious traffic analysis and improving obfuscation while maintaining accountability and transaction privacy, which is important because every blockchain member can see all transactions. Using secure communication protocols between IoT devices or blockchain members can help with this. Systems can also use homomorphic encryption and zero knowledge proofs[15] in some cases, depending on resource availability and IoT devices' technical capabilities.

Proper and logical blockchain implementations based on enforceable smart contracts are essential to a successful large-scale deployment. They improve system performance and minimize blocked transactions that might occur due to a lack of agreement among blockchain members to perform a requested transaction. Integrating legal terms into smart contracts can help enforce participant rules and control misbehavior. This requires techniques that make a generated hash of the legal contract part of the smart contract, which ensures the legal contract's confidentiality.

Smart healthcare applications and services can perform real-time patient monitoring and medical-device actuation, use cloud-based data analytics to improve healthcare quality and the patient experience, and cut costs.

To this end, our flexBeacon system offers a state-of-the-art IoT infrastructure that is agile, flexible, secure, private, and economical. We envision a novel FPGA platform for high

performance, low latency, and the local execution of user-defined beacon and flow rules. We also propose an M2M transceiver and microcontroller for the seamless integration of data for the agile deployment of smart healthcare applications and services.

ACKNOWLEDGEMENTS

This article was made possible by National Priorities Research Program grant no. 7-1113-1-199 from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors.

REFERENCES

1. A. Al-Fuqaha et al., "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communication Surveys & Tutorials*, vol. 17, no. 4, 2015, pp. 2347–2376.
2. S. Datta, C. Bonnet, and N. Nikaein, "An IoT Gateway Centric Architecture to Provide Novel M2M Services," *Proc. 2014 IEEE World Forum on Internet of Things* (WF-IoT 14), 2014, pp. 514–519.
3. J. Treadway, "Using an IoT Gateway to Connect the 'Things' to the Cloud," *TechTarget IoT Agenda*, April 2016; internetofthingsagenda.techtarget.com/feature/Using-an-IoT-gateway-to-connect-the-Things-to-the-cloud.
4. F. Granelli et al., "Software Defined and Virtualized Wireless Access in Future Wireless Networks: Scenarios and Standards," *IEEE Communications Magazine*, vol. 53, no. 6, 2015, pp. 26–34.
5. A. Caraguay et al., "SDN: Evolution and Opportunities in the Development IoT Applications," *Int'l J. Distributed Sensor Networks*, vol. 10, no. 5, 2014, pp. 1–10.
6. *Network Functions Virtualisation (NFV); Use Cases*, ETSI GS NFV 001 v1.1.1, specification by the European Telecommunication Standards Institute, 2013; www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf.
7. "Whole System Demonstrator Programme Headline Findings: December 2011," UK Dept. Health, 5 Dec. 2011; www.gov.uk/government/publications/whole-system-demonstrator-programme-headline-findings-december-2011.
8. *The Healthcare Analytics Adoption Model: A Framework and Roadmap*," white paper, Health Catalyst, 2016; www.healthcatalyst.com/wp-content/uploads/2013/11/analytics-adoption-model-Nov-2013.pdf.
9. M. Boulos et al., "Social Web Mining and Exploitation for Serious Applications: Technosocial Predictive Analytics and Related Technologies for Public Health, Environmental and National Security Surveillance," *Computer Methods and Programs in Biomedicine*, vol. 100, no. 1, 2010, pp. 16–23.
10. D. Mccoy et al., "Shining Light in Dark Places: Understanding the Tor Network," *Proc. 8th Int'l Symp. Privacy Enhancing Technologies* (PETS 08), 2008, pp. 63–76.
11. K. Sakai et al., "An Analysis of Onion-Based Anonymous Routing for Delay Tolerant Networks," *Proc. IEEE 36th Int'l Conf. Distributed Computing Systems* (ICDCS 16), 2016, pp. 609–618.
12. A. Panchenko, F. Lanze, and T. Engel, "Improving Performance and Anonymity in the Tor Network," *Proc. IEEE 31st Int'l Performance Computing and Communications Conf.* (IPCCC 12), 2012, pp. 1–10.
13. G. Zyskind, O. Nathan, and A. Pentland, "Decentralizing Privacy: Using Blockchain to Protect Personal Data," *Proc. 2015 IEEE Security and Privacy Workshops* (SPW 15), 2015, pp. 180–184.
14. M. Condoluci, F. Sardis, and T. Mahmoodi, "Softwarization and Virtualization in 5G Networks for Smart Cities," *Proc. EAI Int'l Conf. Cyber Physical Systems, IoT, and Sensors Networks* (CYCLONE 15), 2015, pp. 2–9.
15. *MultiChain Private Blockchain*, white paper, Coin Sciences Ltd., July 2015; www.multichain.com/download/MultiChain-White-Paper.pdf.

This article originally appeared in Computer, *vol. 50, no. 7, 2017.*

MOHAMMAD A. SALAHUDDIN is a postdoctoral fellow in the University of Waterloo's David R. Cheriton School of Computer Science. Contact him at mohammad.salahuddin@ieee.org.

ALA AL-FUQAHA is a professor in Western Michigan University's Computer Science Department and director of its NEST (Computer Networks, Embedded Systems, and Telecommunications) Research Lab. Contact him at ala.alfuqaha@wmich.edu.

MOHSEN GUIZANI is a professor in and chair of the University of Idaho's Department of Electrical and Computer Engineering. Contact him at mguizani@ieee.org.

KHALED SHUAIB is a professor in the United Arab Emirates University's College of Information Technology. Contact him at k.shuaib@uaeu.ac.ae.

FARAG SALLABI is an associate professor in the United Arab Emirates University's College of Information Technology. Contact him at f.sallabi@uaeu.ac.ae.

The Design and Architecture of Microservices

ALAN SILL

Texas Tech University,
alan.sill@standards-now.org

TO EXPLORE THE ROLE OF DESIGN IN SOFTWARE, CONSIDER TWO OTHER FIELDS THAT ALSO DEPEND ON IT: ART AND ARCHITECTURE. Like art, much of software design can be a matter of taste. As in the art world, issues that inspire passionate debate in the field of software design resonate most loudly within its internal boundaries, and don't necessarily have much of an effect outside of those boundaries.

Design is also important in architecture, both for aesthetic and physically important reasons. As in the structure of buildings, architectural design can have serious ramifications on the reliability, robustness, and suitability for use of software. Like architects, developers are generally aware of the importance of internal structural elements in software, and study and debate the performance and business reasons for selecting one approach over another.

One would not wish to use the plans for a personal home to build a larger structure, such as a sports arena, concert hall, or high-rise building. In the same way, the choice of architectural design pattern in software must be tuned to the desired application, workload, and expected level of use.

Aesthetics and user reaction are important in all of these settings. No one would argue at this stage, in which products from all vendors tend to be beautifully designed, about the need to pay significant attention to issues of user experience and ease of use in software design. Just as we enjoy beautifully designed and functional buildings, software designs are most enjoyable when they're both useful and artfully built.

Microservice Architectures

Concepts related to microservices are discussed extensively elsewhere in this issue. They are, to some degree, old wine in new bottles. The basic approach of separating services into functions that can interact via programming interfaces has been with us for some time. Methods to implement such separation in the framework of service-oriented architectures (SOAs) are also not new.

Recent implementations of microservices in cloud settings, however, take the SOA idea to new limits that are driven by the goals of rapid, interchangeable, easily adapted, and easily scaled components. This is obviously not the only way to use clouds, but it draws well on the basic functional features of cloud computing and is a good match to the corresponding delivery framework. A continued emphasis on the use of RESTful APIs as discussed in previous "Standards Now" columns has also accelerated the pace of change and overall utility of cloud service delivery.

The resulting factorization of workloads and incrementally scalable features of microservices provide a multitude of ways by which SOA can be freed from its previously hidebound, overly formal implementation settings and be implemented in much less forbidding ways. One consequence of this evolution is the development of new architectural patterns and the corresponding emergence and use of standards.

As with art and architecture, much discretion is left to the designer in microservice delivery. You might be tempted to think that standards aren't important, or less important, in the rapidly changing

 Published by the IEEE Computer Society

microservices arena, but this assumption, as I am about to show, wouldn't be correct. To a great degree, the flexibility and ease of implementation of modern approaches to microservice architecture is either compatible with or in fact greatly driven by the emergence of successful design patterns that are already in the process of becoming standards.

Microservice Delivery Using Containers

It would be equally incorrect to equate different trends in cloud design as being equivalent. The current tendency to implement different sets of software in the context of software containers, for example, is more of a coincidence than a direct consequence of microservice design. It's true that containers can be made to isolate execution environments from each other, and that they lend themselves to scalability by allowing such containers to be instantiated quickly on demand.

Other features of software containers require much more work to overcome, however, such as the need to provide well-thought-through mechanisms for network communications between them and associated complications of their use on different physical hosts or on hosts located in different datacenters. Similar problems crop up with regard to security, monitoring, and the need to minimize the operational size of containers. These issues require careful thought and attention to details that aren't directly related to the SOA aspects of microservices themselves.

Despite these shortcomings, microservice delivery matches well in many ways to deployment in software containers. This method is, in fact, currently the most popular way to deploy them, but to deal effectively with the resulting complications just described absolutely requires the use of standards.

This column has covered the emergence of such standards in this area many times, starting with the appc application container specification originally developed by CoreOS, and the runC container engine originally developed by Docker. Much community work has gone into integrating the approaches of these two specification sets and extending them to newer, broader use cases.

Two current relevant projects of the Linux Foundation are the Open Container Initiative (www.opencontainers.org) and the Cloud Native Computing Foundation (CNCF, https://cncf.io). Popular image formats include ACI, the container image format defined in the appc specification, and OCI, the Open Containers Image Format specification. Much of the work going on within the CNCF is aimed higher up in the software stack and addresses the large-scale behaviors of distributed systems of microservices.

Although work is still in progress on various aspects of each of these standards within these organizations and on their relationship to each other, it's encouraging to see efforts of this sort emerge naturally from ongoing community interests.

> Just as we enjoy beautifully designed and functional buildings, software designs are most enjoyable when they're both useful and artfully built.

Data Formats and APIs

To make microservice architectures work in practice, one must get information into and out of these services and find ways to make the information exchange and control-passing features take place at component boundaries. Programmers must therefore address design topics dealing with data exchange and messaging, and must implement these services with suitable orchestration and control.

Standards exist that provide the basis for such data exchange. The most popular data formats in cloud computing are JavaScript Object Notation (JSON) and XML. JSON is documented in two standards: Ecma International's ECMA-404[1] and IETF's RFC 7159.[2] XML is a somewhat older but still popular text-based format for data exchange supported by several W3C standards. Although it isn't as human-readable as JSON, each format has particular strengths and weaknesses and both are still very much in use.

For Internet of Things (IoT) and sensor-oriented settings, as discussed in the previous issue on manufacturing, the Sensor Network Object Notation (SNON, www.snon.org) is a representation based on JSON that includes some predefined fields that are especially useful in dealing with sensor data. In addition, the Data Distribution Service (DDS, http://www.omg.org/spec/DDS) and DDS Data Local Reconstruction Layer (DDS-DLRL) specifications were developed by the Object Management Group specifically to handle data interchange tasks related to IoT systems.

> General data standards are available to deal with the wide variety of formats for datasets without having to be locked into a particular format.

Unlike the other protocols I've mentioned, DDS can handle content-aware network routing, data prioritization by transport priorities, and both unicast and multicast communications within the methods defined by the standard set itself.

Additionally, general data standards are available to deal with the wide variety of formats for datasets without having to be locked into a particular format. For example, working with the US National Center for Supercomputing Applications (NCSA, http://www.ncsa.illinois.edu) and IBM, the Open Grid Forum has developed a language for describing the structure of data formats without needing to rewrite them. The resulting Data Format Description Language (DFDL, www.ogf.org/dfdl) is a flexible, general specification set suited to a wide variety of data input, output, and format transcription problems and is supported by both commercial and open source software implementation tools.

Many approaches currently used in microservices create custom APIs for access to specific data. This approach is compatible with, though typically implemented without, reference to external data format standards. As a result, current cloud microservice designs are burdened with a huge variety and multiplicity of API definitions.

In previous columns, I've referred to the API directory maintained, for example, by the website ProgrammableWeb.com, which at the time of this writing maintains a directory of more than 15,000 APIs (www.programmableweb.com/apis/directory). This situation requires APIs to be designed to work either in small subsets of the application arena in which either the API is stable, or to be built to a common self-describing or standardized pattern.

Examples of effective API standards are the RESTful API Markup Language (RAML, http://raml.org) and Swagger, which has evolved into the Open API Initiative (https://openapis.org), as discussed in previous columns.

Messaging Standards

The next step after understanding data formats and APIs for data exchange is to move in the direction of messaging and application control. HTTP and its secure variant HTTPS are the most familiar messaging standards, and are specified in a range of IETF documents summarized at the working group website (httpwg.org/specs).

The IETF specifications underlying TCP form the basis of a large fraction of Internet traffic. TCP continues to receive detailed attention from the community due to its importance in a wide variety of settings. The most important TCP specifications and their relationships with one another are summarized in RFC 7414.[3] A number of other application, transport Internet, and link layer protocols are also useful.[4]

The User Datagram Protocol (UDP) is useful for Internet communications that can be intermittent or don't have to be completely received at all times.[5] UDP can be used to carry out IP communications in situations in which handshaking and verification of receipt of the individual message packets aren't necessary. The Stream Control Transmission Protocol (SCTP) provides an alternative to TCP and UDP applicable to streaming use cases.[6]

Another example of a manufacturing-relevant specialized transfer standard is the Constrained Ap-

plication Protocol (CoAP).[7] According to its description, "CoAP provides a request/response interaction model between application endpoints, supports built-in discovery of services and resources, and includes key concepts of the Web such as URIs and Internet media types. CoAP is designed to easily interface with HTTP for integration with the Web while meeting specialized requirements such as multicast support, very low overhead, and simplicity for constrained environments."

The Extensible Messaging and Presence Protocol (XMPP) is an XML-based communications standard designed for message-oriented middleware communications. The core specifications for XMPP are RFCs 6120,[8] 6121,[9] and 7622[10] and include a WebSocket binding defined in RFC 7395.[11] Several extensions beyond the base specifications are supported by the dedicated XMPP organization (see http://xmpp.org/extensions). Beyond its applications to human-oriented communications, XMPP is also used in smart electrical grid applications and a variety of industrial settings. Several extensions oriented toward use in IoT settings were published in late 2015.

Methods to handle publish/subscribe messaging can have advantages compared to the protocols when used for machine-to-machine communications at high speeds. The Message Queuing Telemetry Transport (MQTT, http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html), recently standardized by the Organization for Advanced Structured Information Systems (OASIS), is another example of such a method.

The Advanced Message Queuing Protocol (AMQP) is another popular middleware messaging standard set. It can be applied using either publish/subscribe or point-to-point communication patterns. OASIS published AMQP as a set of standards in 2014 (www.oasis-open.org/standards#amqpv1.0) and adopted it as a joint International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) later that year.[12] AMQP has a layered architecture, and the specification set is organized into different parts to reflect that architecture.

Networking Considerations

Networking provides the core feature that ties all cloud services to each other. I discussed this topic extensively in the May/June 2016 issue of this magazine,[4] stating there that "the underpinnings of the cloud consist of the ways in which otherwise disconnected, highly scaled, and rapidly changing collections of service components can be instantiated and hooked together swiftly and flexibly to form the basis of a cloud service."

The need for performance is especially important in the implementation of microservice architectures. This consideration obviously provides the practical limit to the degree to which individual service components can be scaled down in terms of information exchange and functionality. Issues related to security, connectivity between microservice components, and scalability also have considerations that are affected by the choice of networking architecture and protocols.

A US National Institute of Standards and Technology draft publication covers this topic, with an emphasis on security considerations.[13] Although the comment period has closed on this particular draft, the topic's general nature makes it seem to me that the issues discussed in this document will be revisited several times in the near future as the general area of microservice delivery matures.

Special considerations that relate to networking are also pushing some microservice frameworks in directions that lead away from human readability of the interchanged data and even of the on-the-wire protocols used in the API calls and messaging. Examples that I've discussed in previous issues continue to mature, including the recent release by Google of its gRPC framework at version 1.0 (https://github.com/grpc/grpc/releases/tag/v1.0.0) after an extended period of development, with multiple language bindings now available.

The design approach underpinning gRPC makes extensive use of protocol buffers (https://developers.google.com/protocol-buffers/docs/reference/overview), a design construct intended to serialize structured data in a simpler manner than in XML but without some of JSON's limitations, and is designed to be compatible with HTTP/2. My personal belief is that these developments illustrate the emergence of new design trends for cloud services that favor speed and responsiveness over human readability of the exchanged data and API calls, and that might lead to radical revisions of some of the fundamental assumptions that govern microservices in the future.

This article originally appeared in
IEEE Cloud Computing, *vol. 3, no. 5, 2017.*

THE DISCUSSION HERE HAS FOCUSED ON THE DESIGN AND ARCHITECTURE OF MICROSERVICES. I've covered considerations related to packaging and delivery of microservices in containers, data exchange, and data formats, messaging and networking, focusing on some up-to-date topics on standards related to these areas.

My next column will address topics related to microservices orchestration, including relevant standards such as Topology and Orchestration Specification for Cloud Applications (Tosca) and Cloud Application Management for Platforms (CAMP); microservices control, including the Open Cloud Computing Interface (OCCI) and Cloud Infrastructure Management Interface (CIMI) standard sets; and serverless microservices, such as Amazon Lambda and related concepts. I'll also take another look at the SOA basis for microservice architectures to tie both of these columns together.

As always, this discussion only represents my own viewpoint. I'd like to hear your opinions and experience in this area. I'm sure other readers of the magazine would also appreciate additional information on this topic.

Please respond with your input on this or previous columns. Please include news you think the community should know in the general areas of cloud standards, compliance, or related topics. I'm happy to review ideas for potential submissions to the magazine or for proposed guest columns. I can be reached for this purpose at alan.sill@standards-now.org.

References

1. Ecma International, *The JSON Data Interchange Format*, Ecma-404, 1st ed. 2013; www.ecma-international.org/publications/standards/Ecma-404.htm.
2. T. Bray, ed., *The JavaScript Object Notation (JSON) Data Interchange Format*, IEEE RFC 7159, 2014; https://www.rfc-editor.org/info/rfc7159.
3. M. Duke, et al., *A Roadmap for Transmission Control Protocol (TCP) Specification Documents*, IETF RFC 7414, 2015; www.rfc-editor.org/info/rfc7414.
4. A. Sill, "Standards Underlying Cloud Networking," *IEEE Cloud Computing*, vol. 3, no. 3, 2016, pp. 76–80.
5. J. Postel, *User Datagram Protocol*, IETF RFC 768, 1980; www.rfc-editor.org/info/rfc768.
6. R. Stewart, ed., *Stream Control Transmission Protocol*, IETF RFC 4960, 2007; www.rfc-editor.org/info/rfc4960.
7. Z. Shelby, K. Hartke, and C. Bormann, *The Constrained Application Protocol (CoAP)*, IETF RFC 7252, 2014; www.rfc-editor.org/info/rfc7252.
8. P. Saint-Andre, *Extensible Messaging and Presence Protocol (XMPP): Core*, IETF RFC 6120, 2011; www.rfc-editor.org/info/rfc6120.
9. P. Saint-Andre, *Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence*, IETF RFC 6121, 2011; www.rfc-editor.org/info/rfc6121.
10. P. Saint-Andre, *Extensible Messaging and Presence Protocol (XMPP): Address Format*, IETF RFC 7622, 2015; www.rfc-editor.org/info/rfc7622.
11. L. Stout, ed., *An Extensible Messaging and Presence Protocol (XMPP) Subprotocol for WebSocket*, IETF RFC 7395, 2014; www.rfc-editor.org/info/rfc7395.
12. *Information technology—Advanced Message Queuing Protocol* (AMQP), Int'l Organization for Standardization/Int'l Electrotechnical Commission, ISO/IEC 19464, v.1.0, 2014; www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=64955.
13. A. Karmel, R. Chadromouli, and M. Iorga, "NIST Definition of Microservices, Application Containers and System Virtual Machines," Nat'l Inst. of Standards and Technology (NIST) Special Publication 800-180, 2016; http://csrc.nist.gov/publications/drafts/800-180/sp800-180_draft.pdf.

ALAN SILL *directs the US National Science Foundation's Cloud and Autonomic Computing industry/university cooperative research center. He's interim senior director of the High Performance Computing Center and adjunct professor of physics at Texas Tech University, and visiting professor of distributed computing at the University of Derby. Sill has a PhD in particle physics from American University. He's an active member of IEEE, the Distributed Management Task Force, and the TeleManagement Forum, and he serves as president for the Open Grid Forum. He's a member of several cloud standards working groups and national and international standards roadmap committees, and he remains active in particle physics and advanced computing research. Contact him at alan.sill@standards-now.org.*

COMPSAC 2018

Tokyo, Japan

July 23-27

Staying Smarter in a Smartening World

IEEE Advancing Technology for Humanity

IEEE computer society

Call for Papers

COMPSAC is the IEEE Computer Society Signature Conference on Computers, Software and Applications. It is a major international forum for academia, industry, and government to discuss research results and advancements, emerging challenges, and future trends in computer and software technologies and applications. The theme of COMPSAC 2018 is Staying Smarter in a Smartening World.

Computer technologies are producing profound changes in society. Emerging developments in areas such as Deep Learning, supported by increasingly powerful and increasingly miniaturized hardware, are beginning to be deployed in architectures, systems, and applications that are redefining the relationships between humans and technology. As this happens, humans are relinquishing their roles as masters of technology to partnerships wherein autonomous, computer-driven devices become our assistants. What are the technologies enabling these changes? How far can these partnerships go? What will be our future as we deploy more and more "'things" on the Internet of Things – to create smart cities, smart vehicles, smart hospitals, smart homes, smart clothes, etc.? Will humans simply become IoT devices in these scenarios and if so, what will be the social, cultural, and economic challenges arising from these developments? What are the technical challenges to making this all happen – for example, in terms of technologies such as Big Data, Cloud, Fog, Edge Computing, mobile computing, and pervasive computing in general? What will be the role of the 'user' as the 21st Century moves along?

COMPSAC 2018 is organized as a tightly integrated union of symposia, each of which will focus on technical aspects related to the "smart" theme of the conference. The technical program will include keynote addresses, research papers, industrial case studies, fast abstracts, a doctoral symposium, poster sessions, and workshops and tutorials on emerging and important topics related to the conference theme. A highlight of the conference will be plenary and specialized panels that will address the technical challenges facing technologists who are developing and deploying these smart systems and applications. Panels will also address cultural and societal challenges for a society whose members must continue to learn to live, work, and play in the environments the technologies produce. Authors are invited to submit original, unpublished research work, as well as industrial practice reports. Simultaneous submission to other publication venues is not permitted. All submissions must adhere to IEEE Publishing Policies, and all will be vetted through the IEEE CrossCheck Portal.

Standing Committee Chair: Sorel Reisman, California State University, USA
Steering Committee Chair: Sheikh Iqbal Ahamed, Marquette University, USA

General Chairs: Shinichi Honiden (NII, Japan)
Roger U. Fujii, Fujii Systems, 2016 IEEE Computer Society Prelident

Program Chairs in Chief:
Jiannong Cao (Hong Kong Polytechnic University, Hong Kong)
Stelvio Cimato (University of MIlan, Italy)
Yasuo Okabe (Kyoto University, Japan)
Sahra Sedighsarvestani (Missouri University of Science & Technology, USA)

Workshop Chairs: Kenichi Yoshida (University of Tskuba, Japan)
Ji-Jiang Yang (Tsinghua University, China)
Hong Va Leong (Hong Kong Polytechnic University, Hong Kong)
Chung Horng Lung (Carleton University, Canada)

Local Chair: Hironori Washizaki (Waseda University, Japan)

Important Dates

Workshop proposals
Due date: 15 October 2017
Notification: 15 November 2017

Main Conference papers
Due date: 15 January 2018
Notification: 31 March 2018

Workshop papers
Due date: 10 April 2018
Notification: 1 May 2018

Camera Ready and Registration
Due date: May 15, 2018

Next-Generation Mobile Services

M. Brian Blake • *Drexel University*

As a preteen/teen in the early to mid-80s, I was an avid subscriber of comic books. My most-purchased comic book was *The Amazing Spiderman*, as Spiderman was my favorite superhero. However, being African-American, I was also fond of the African-American super heroes, which at the time included James Rhodes — who served as Iron Man for a stint and Luke Cage (also known as Power Man) from the *Power Man and Iron Fist* comics. For those who aren't familiar with the comic book series from 30 years ago, then you might have seen the recent Netflix shows that have created separate *Luke Cage* and *Iron Fist* television series. At this point, you might be wondering about the relationship of comic books to the title of this editorial.

A character in the *Power Man and Iron Fist* comic book series was Gordy, who was an agent of the Special Military Intelligence Law Enforcement division (S.M.I.L.E; www.marvunapp.com/Appendix4/gordysmile.htm#smile). Gordy's weapon of choice was his mobile device. His early '80s cellular phone, modeled as a household cordless phone, had the ability to create force fields in addition to projecting particle beams. The phone could also be used as a boomerang. On occasion he did actually make regular phone calls with it. Interestingly enough, in the early '80s, they didn't mention him using it to access the Internet.

So, how does Gordy's weapon tie in with the latest trends at the intersection of mobile computing and Internet computing? Well, much like Gordy's use of mobile devices and generally all other areas of computing, a popular concern is the interaction of information and functions on the Internet with the real world using the mobile device as a medium. There are several themes that span next-generation mobile services as it relates to Internet computing.

Sensing and crowdsensing will enhance Internet computing. The use of mobile devices, and particularly those with location-based services, has become pervasive. The wide distribution of mobile users with GPS, accelerometers, cameras, and real-time human conversations has made it possible to provide situational awareness of the real world in areas such as traffic congestion, urban networks, and Wi-Fi conditions, as well as other geographical information services.[1] There are several challenges that Internet computing experts must address, to name a few:

- There must be approaches to process big data from end-user mobile devices to mobile service providers. Subsequently, this information must be routed through the network to web services that can process the information within service-side applications.
- Other approaches must define authoritative sources in real time. Sensor-based information from multiple users representing the same or proximal situations will need to be merged, and in some cases conflicts must be resolved.
- The need to merge information and functions about real-world situations might lead to the resurgence of the techniques for web service discovery and composition. Discovering related functions on the Internet to fuse user-supplied information with information provided by web services might be the best way to ensure that the best operational picture is constructed in real time.

Accessible augmented or virtual reality will integrate with and leverage mobile devices. The use of augmented reality is currently (re)gaining

 Published by the IEEE Computer Society

traction when intersecting mobile computing with Internet computing: for example, with the introduction or reintroduction of augmented reality headsets such as Google Glass[2] and the recent virtual reality implemented with Samsung Galaxy.[3] Recent studies are investigating the use of augmented reality headsets to connect to mobile devices,[4] particularly as a user interface enabling access to large, multidimensional data sources. Many of the previously mentioned challenges of merging personal and public information must be used to create a subset of the most relevant information to fit for display on these devices. Moreover, the infrastructure to support these connections must intelligently queue information both on the Internet servers as well as the limited space on mobile devices.

As you can see, with the re-emergence of Power Man and Iron Fist 35 years later, our present-day Gordys might have other futuristic techniques for mobile devices beyond the boomerang and particle beams.

And in that vein, I hope you enjoy this month's special issue on 5G. The guest editors — N.K. Shankaranarayanan and Arunabha Ghosh — along with the article authors and reviewers, have done a great job representing an emerging area of technology.

References

1. M. Xiao et al., "Online Task Assignment for Crowdsensing in Predictable Mobile Social Networks," *IEEE Trans. Mobile Computing*, vol. 16, no. 8, 2017, pp. 2306–2320.
2. C. Merriman, "Your Pupil Is About to Become the Master: Google Glass Is Coming Back," *The Inquirer*, 7 July 2017; www.theinquirer.net/inquirer/news/3013396/your-pupil-is-about-to-become-the-master-google-glass-is-coming-back.
3. D. Bosnjak, "New Samsung Galaxy Note 8 Leak Hints at AR Capabilities," *Android Headlines*, 7 July 2017; www.androidheadlines.com/2017/07/new-samsung-galaxy-note-8-leak-hints-ar-capabilities.html.
4. S. Lin et al., "Ubii: Physical World Interaction Through Augmented Reality," *IEEE Trans. Mobile Computing*, vol. 16, no. 3, 2017, pp. 872–885.

M. Brian Blake is the provost and executive vice president of academic affairs at Drexel University. As a professor of computer science and electrical engineering, his research interests are in service-oriented computing, adaptive distributed systems, and Web-based software engineering. Blake has a PhD in information and software engineering from George Mason University. Contact him at mbrian.blake@drexel.edu.

This article originally appeared in IEEE Internet Computing, *vol. 21, no. 5, 2017.*

Human Augmentation

Editor: Albrecht Schmidt ■ University of Stuttgart ■ albrecht@computer.org

Superhuman Sports: Applying Human Augmentation to Physical Exercise

Kai Kunze and Kouta Minamizawa, *Keio University*
Stephan Lukosch, *Delft University of Technology*
Masahiko Inami and Jun Rekimoto, *Tokyo University*

Have you ever imagined flying through the sky like a bird, climbing walls like a spider, or playing fictional sports—such as Quidditch in the *Harry Potter* books? When we're young, we often role-play and pretend we have superpowers. Inspired by these experiences, we started working on what we refer to as *superhuman sports*, focusing on how we might make some of these powers a reality.

We want to create an application area to explore human augmentation and enhance human abilities in a playful way. The field of superhuman sports combines competition and physical elements from traditional sports with technology to overcome the somatic and spatial limitations of our human bodies. The field serves as a fascinating application area for human augmentation.

TOWARD SUPERHUMAN SPORTS

Science—in particular, information technology—is already an integral part of today's sports training and events. However, traditional sports emphasize the achievements of the individual. Sport federations and competitions struggle when it comes to knowing how to deal with the augmented human concept.

Consider, for example, Markus Rehm, an amputated long jumper with a prosthetic right leg. He uses a blade-type leg prosthesis when competing. In 2014, he qualified for the European Championships but wasn't allowed to compete,[1] because his prosthesis was viewed as a violation of the rules, even though it hasn't been proven that the prosthesis gives him a natural advantage; other athletes with similar prostheses didn't perform so well.[2] This illustrates the current direction and challenges of allowing augmented humans to participate in conventional sports.

In contrast, superhuman sports aim to create a field where people compete, overcoming technological—rather than solely human—limitations. The focus is on improving cognitive and physical functions of the human body, creating artificial senses and reflexes to participate in sports competitions. We want to create and explore new experiences with these novel senses and reflexes by augmenting old sports and designing new ones, enhancing sports training, and sharing the experiences with both local and remote audience members.

Superhuman sports exploit human augmentation, using technology to surpass the physical and cognitive restrictions of our bodies and enabling superhuman senses and abilities. The core concepts of superhuman sports include augmenting the body, playing field, training opportunities, and even spectators.

EXAMPLES AND EXPERIENCES

Superhuman sports researchers span a few different disciplines, but most are focused on the fields of augmented and virtual reality, wearable computing, and human-computer interaction. Here, we present some example technologies from these areas.

Body Augmentation

Augmenting the body is the most straightforward notion. The goal is to enhance a sports practitioner's inherent abilities using wearable technologies and implantables.

For example, Skeletonics lets the user climb into a completely mechanical exoskeleton so that he or she can enjoy a different body model and new perspective (see Figure 1a). Bubble Jumper (or Bubble Sumo) deploys a combination of skyrunner stilts and a bubble ball around the player's torso (see Figure 1b). The goal of the sport is to knock over the other player.

Field or Sport Augmentation

Augmenting the playing field aims to make the sports more interesting and enjoyable to play—for example, using projection technologies to indicate where a play ball might go or converting part of a sports field into a virtual ocean. A representative case that one of us (Inami) has worked on is AquaCave,

 Published by the IEEE Computer Society

(a) (b)

Figure 1. Examples of body augmentation: (a) Skeltonics (http://skeletonics.com) offers mechanical exoskeletons for entertainment, while (b) Bubble Jumper lets participants use stilts and a bubble ball to knock over the opponent.

(a) (b)

Figure 2. An example of a mixed reality sport: (a) Hado and (b) HadoKart, which moves the mixed reality sport into a motor kart scenario.

which enhances the swimming experience by surrounding the swimmer with rear-projection acrylic walls, providing an immersive stereoscopic projection environment.[3]

However, such augmentations can also lead to entirely new sports. One example is Hado (http://meleap.com), where players compete against each other using a head-mounted display for augmented reality and a gesture armband to detect muscle activity and arm movements (see Figure 2). Hado is already a commercial sport in Japan, marketed by Meleap Inc. These mixed-reality experiences serve as an inspiration to create new sports.

Another example that one of us (Inami) has worked on, involving more toward state-of-the-art research, is SpiderVision,[4] a wearable device that can extend the field of human vision. SpiderVision merges the view from front and back cameras on a VR headset to inform users about activities happening behind them (see Figure 3). Having 360-degree vision is a valuable skill for any complex team sport (from football to synchronized swimming), where formation and relative positioning of players to each other matters.

Training Augmentation

Augmented training deploys information technology to improve training and enhance the inherent capabilities of professional and amateur sports practitioners—for example, using electric muscle stimulation to build up specific muscle regions or transcranial direct current stimulation to improve hand-eye coordination or other motor tasks.

Another project one of us (Kunze) is exploring aims to alter the clues in our

HUMAN AUGMENTATION

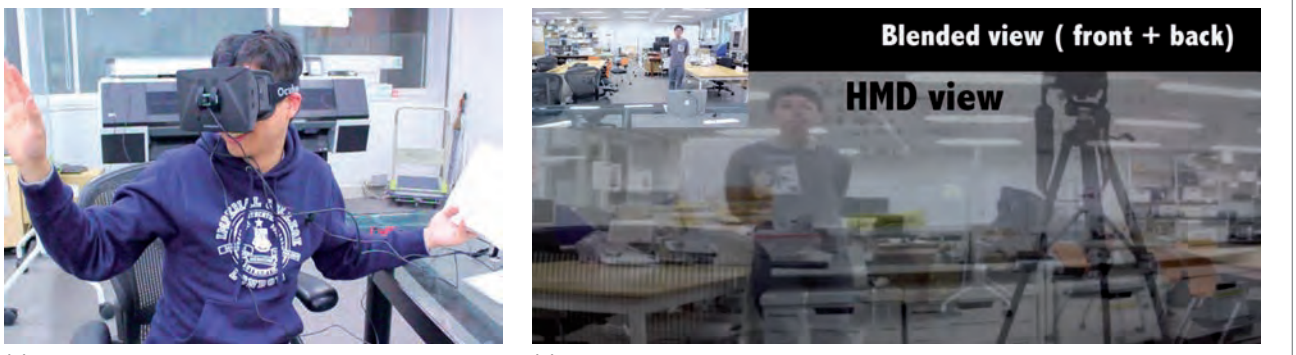


Figure 3. SpiderVision: (a) the VR headset with rear and front cameras can (b) blend the view of both for the user.

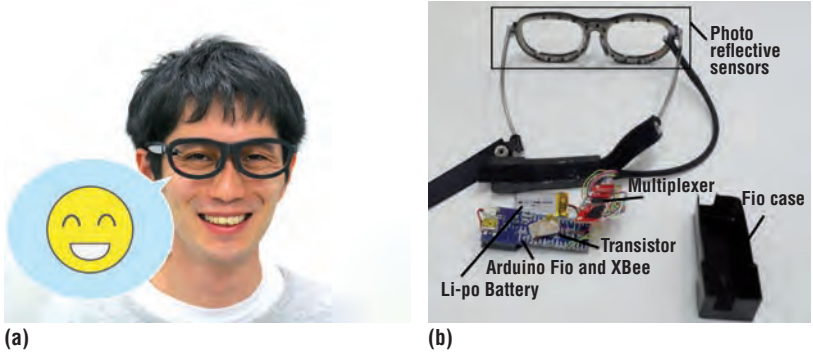


Figure 4. AffectiveWear: tracking facial expression of sport spectators using smart glasses.[8]

perception using smart glasses that influence our motions using projections in our peripheral vision.[5] So far, the smart glasses apply the principle of a "vection field" to alter the speed of a person walking. Future prototypes might also encourage slight changes in direction. To give a couple of examples, the technology could be used during running competitions to regulate and find the optimal pace of an athlete. In training situations, we could teach athletes to perform movements (such as changes in direction) with precise timing.

More invasive and futuristic is the use of galvanic vestibular stimulation (GVS) to alter a practitioner's movements. GVS stimulates the vestibular system using a weak current behind the user's ear. This influences the sense of equilibrium so the user feels like falling toward the anode. GVS can be used to "steer" people and influence the user's perception of speed.[6] This technology can augment any sport relying on a sense of balance or velocity. Take skiing as an example—applying GVS in a training situation could better stabilize the athletes and help them determine the best body posture relative to the ski slope.

Spectator Augmentation

Augmenting "cheering" focuses on new experiences for those watching a sporting event. For example, those in the crowd might feel the adrenaline rush of an athlete before scoring an important point, or they might experience the exhaustion of a marathon runner just before he or she crosses the finish line.

JackIn Head is a new device that one of us (Rekimoto) is working on that has given users an immersive 360-degree camera view from the sport practitioner's perspective, with minimal setup.[7]

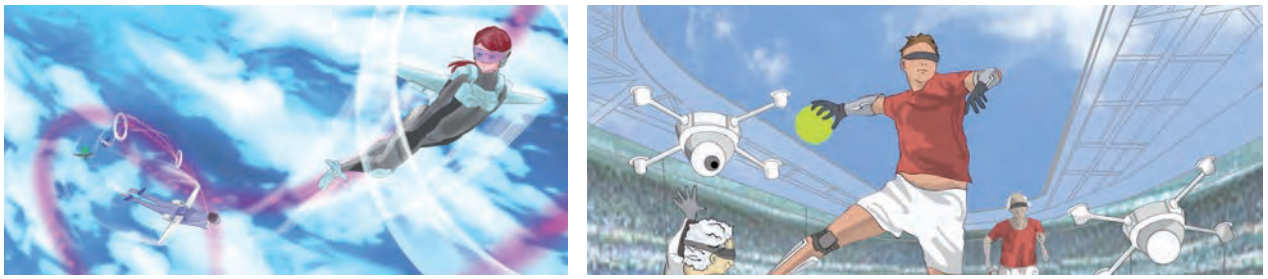
Moving away from sharing basic vision and sound, new technologies can help share the "affect" of the sport. For example, two of us (Kunze and Inami) are working on AffectiveWear smart glasses, which can detect the user's facial expressions by monitoring the distance between glasses frame and face using photo-reflective sensors (see Figure 4).[8] AffectiveWear can aggregate the facial expressions of spectators to help organizers evaluate a sporting event and to offer a more crowd-like experience for home viewers with virtual cheers.

On another level, haptic feedback can provide more immersion while watching sports competitions. For instance, the Synesthesia Suit (a project Minamizawa is researching) gives an immersive embodied VR experience with 24 vibro-tactile actuators distributed over the entire body.[9]

LOOKING TOWARD THE 2020 OLYMPICS

To promote the concept of superhuman sports, we founded the Superhuman Sports Society in Japan in 2015. The early activities included ideation workshops with designers, art schools, and the general public working on concepts for novel sports (see Figure 5). In the next stage, we're holding workshops and hackathons to test technologies and create experiences. We already successfully held a Superhuman Sports Expo and several Superhuman Sports Games events, where the broader public tried

This article originally appeared in IEEE Pervasive Computing, *vol. 16, no. 2, 2017.*



(a) (b)

Figure 5. Working on concepts for novel sports—two pieces from ideation workshops: (a) fusing the human dream of flying with a competitive superhuman sport and (b) augmented sports involving all people disregarding age, physical ability, and cognitive skills overcoming our bodies' limitations.

out the novel sport designs, including Bubble Jumper and Hado.

For Tokyo 2020, we're creating a superhuman sports culture. Through these efforts, we're not only creating new sports and augmenting players, fields, training opportunities, and spectators; we're also educating creators of future superhuman sports.

The next steps include preparing for the first superhuman sports tournament this year and moving slowly from amateur to professional players. For 2018, a superhuman sports competition is scheduled to be included in the National Sports Festival in Fukui, and we're in preparations to host a Superhuman Design Competition at TU Delft at the end of 2019 (contact us at s.g.lukosch@tudelft.nl if you're interested in participating). Finally, in 2020, we plan to hold a National Superhuman Sports Championship, and we eventually hope to make such events international.

ACKNOWLEDGEMENTS

Some of the authors' research has been supported by JST Presto. We also thank the Leibniz Center Dagstuhl for fruitful discussions around enabling technologies shaping novel research directions at the Dagstuhl Seminar "Beyond VR and AR" (www.dagstuhl.de/17062).

REFERENCES

1. L. Greenemeier, "Blade Runners: Do High-Tech Prostheses Give Runners an Unfair Advantage?" *Scientific Am.*, 5 Aug. 2016; www.scientificamerican.com/article/blade-runners-do-high-tech-prostheses-give-runners-an-unfair-advantage.
2. J.-H. Raffler, "Do Prosthetic Legs Provide Unfair Advantage?" *Deutsche Welle*, 30 July 2014; http://dw.com/p/1Cm86.
3. S. Yamashita, X. Zhang, and J. Rekimoto, "AquaCAVE: Augmented Swimming Environment with Immersive Surround-Screen Virtual Reality," *Proc. 29th Ann. Symp. User Interface Software and Technology*, 2016, pp. 183–184.
4. K. Fan et al., "SpiderVision: Extending the Human Field of View for Augmented Awareness," *Proc. 5th Augmented Human Int'l Conf.*, 2014, article no. 49.
5. T. Nakuo and K. Kunze, "Smart Glasses with a Peripheral Vision Display," *Proc. 2016 ACM Int'l Joint Conf. Pervasive and Ubiquitous Computing: Adjunct* (UbiComp), 2016, pp. 341–344.
6. N. Nagaya et al., "Visual Perception Modulated by Galvanic Vestibular Stimulation," *Proc. 2005 Int'l Conf. Augmented Tele-Existence*, 2005, pp. 78–84.
7. S. Kasahara and J. Rekimoto, "JackIn: Integrating First-Person View with Out-of-Body Vision Generation for Human-Human Augmentation," *Proc. 5th Augmented Human Int'l Conf.*, 2014, article no. 46.
8. K. Masai et al., "AffectiveWear: Toward Recognizing Facial Expression," *Proc. ACM SIGGRAPH 2015 Emerging Technologies*, 2015, article no. 4.
9. Y. Konishi et al., "Synesthesia Suit: The Full Body Immersive Experience," *Proc. ACM SIGGRAPH 2016 VR Village*, 2016, article no. 20.

Kai Kunze is an associate professor at the Keio Graduate School of Media Design, Keio University, Tokyo Japan. Contact him at kai@kmd.keio.ac.jp.

Kouta Minamizawa is an associate professor at the Keio Graduate School of Media Design, Keio University, Tokyo Japan. Contact him at kouta@kmd.keio.ac.jp.

Stephan Lukosch is an associate professor at the Delft University of Technology. Contact him at s.g.lukosch@tudelft.nl.

Masahiko Inami is a professor at the Research Center for Advanced Science and Technology, University of Tokyo. Contact him at inami@inami.info.

Jun Rekimoto is a professor at the Interfaculty Initiative in Information Studies, The University of Tokyo. Contact him at rekimoto@acm.org.

EDITORS: Rick Kuhn, NIST, kuhn@nist.gov
Tim Weil, Scram Systems, tweil.ieee@gmail.com

Can Blockchain Strengthen the Internet of Things?

Nir Kshetri, *University of North Carolina at Greensboro*

Blockchain—a kind of distributed ledger technology—has been described in the popular press as the next big thing. Put simply, a blockchain is a data structure that makes it possible to create a tamper-proof digital ledger of transactions and share them. This technology uses public-key cryptography to sign transactions among parties. The transactions are then stored on a distributed ledger. The ledger consists of cryptographically linked blocks of transactions, which form a blockchain (bit.ly/2sgabnq). It is impossible or extremely difficult to change or remove blocks of data that are recorded on the blockchain ledger.

Regarding the question of whether blockchain can strengthen the Internet of Things (IoT), the answer—based on this research—is "maybe." Observers have noted that the blockchain–IoT combination is powerful and is set to transform many industries.[1] For instance, IoT devices can carry out autonomous transactions through smart contracts.[2] Combined with artificial intelligence (AI) and big data solutions, more significant impacts can be produced.

A natural question is thus what roles can blockchain play in strengthening IoT security? To demonstrate this problem's significance, consider the following example. In October 2016, the US-based DNS provider Dyn faced cyberattacks. Dyn said the attacks originated from "tens of millions of IP addresses,"[3] and at least some of the traffic came from IoT devices, including webcams, baby monitors, home routers, and digital video recorders.[4] These IoT devices had been infected with malware called Mirai, which controls online devices and uses them to launch distributed denial-of-service (DDoS) attacks. The process involves phishing emails to infect a computer or home network. Then the malware spreads to other devices, such as DVRs, printers, routers, and Internet-connected cameras employed by stores and businesses for surveillance.[5]

From a security standpoint, a main drawback of IoT applications and platforms is their reliance on a centralized cloud. A decentralized, blockchain-based approach would overcome many of the problems associated with the centralized cloud approach. Some point out that blockchain could provide military-grade security for IoT devices.[6] There is no single point of failure or vulnerability in blockchain, except with the clock needed for time stamping.

Considering these observations, this column provides insights into ways in which blockchain might strengthen IoT security.

Incorporating Blockchain into IoT Security

Blockchain's incorporation into IoT is being supported through a wide variety of measures intended to strengthen security. Several companies are leading initiatives to integrate blockchain into their production and supply chains. For instance, IBM is using its large cloud infrastructure to provide

 Published by the IEEE Computer Society

blockchain services for tracking high-value items as they move across supply chains.

The IBM Watson IoT Platform's built-in capability also allows users to add selected IoT data to private blockchain ledgers that can be included in shared transactions. The platform translates the data from connected devices into the format that blockchain contract APIs need. It is not necessary for the blockchain contract to know the specifics of the device data. The platform filters device events and sends only the data that is required to satisfy the contract (ibm.co/2rJWCPC). All business partners can access and supply IoT data in a decentralized fashion and can verify each transaction.[7] Data is not collected, stored, or managed centrally. Rather, it is protected and shared among only the parties involved in the transaction.

Startups such as Provenance use blockchain to promote trust in the supply chain by providing transparency and visibility when the product moves from the source to the customer.[8] Others are creating new business models that eliminate the need for centralized cloud servers. For example, Filament, a blockchain-based solutions provider for IoT, has launched wireless sensors, called Taps, that allow communication with computers, phones, or tablets within 10 miles (bit.ly/2rsxZYf).

Taps create low-power, autonomous mesh networks that enable companies to manage physical mining operations or water flows over agricultural fields. Taps don't rely on cloud services. Device identification and intercommunication is secured by a blockchain that holds the unique identity of each participating node.[9] One key application is likely to be in the next generation of the industrial network (the Industrial Internet). Filament's blockchain-based applications involve sensors connected in a decentralized system and use autonomous smart contracts. This means that devices communicate securely with each other, exchange values, and execute actions automatically. For instance, Filament's Tap can be attached to drilling rigs in remote locations. Based on predefined conditions, a rig might know that it requires a piece of machinery and thus might send a request to an autonomous drone.[10]

Measures are also taken at interorganizational levels. A group of technology and financial companies have announced that they have formed a group to set a new standard for securing IoT applications using blockchain. Companies joining the group include Cisco, Bosch, Bank of New York Mellon, Foxconn Technology, Gemalto, and blockchain startups Consensus Systems, BitSE, and Chronicled.[11] This group hopes to establish a blockchain protocol to build IoT devices, applications, and networks.[12]

Blockchain-based identity and access management systems can be leveraged to strengthen IoT security.

Identity and Access Management Systems

Blockchain-based identity and access management systems can be leveraged to strengthen IoT security. Such systems have already been used to securely store information about goods' provenance, identity, credentials, and digital rights. As long as the original information entered is accurate, blockchain's immutability can be achieved.[13] In this regard, a key challenge that arises in some applications is that it is difficult to ensure that the properties of physical assets, individuals (credentials), resource use (energy and bandwidth through IoT devices), and other relevant events are stored securely and reliably. This aspect can be handled relatively easily for most IoT devices. For instance, a private blockchain can be used to store cryptographic hashes of individual device firmware. Such a system creates a permanent record of device configuration and state. This record can be used to verify that a given device is genuine and that its software and settings have not been tampered with or breached. Only then is the device allowed to connect to other devices or services.

Returning to the Dyn example, IP spoofing attacks were launched for the later versions of the Mirai botnet. Blockchain-based identity and access management systems can provide stronger defense against attacks involving IP spoofing or IP address forgery. Because it is not possible to alter approved blockchains, it is not possible for devices to connect to a network by disguising themselves by injecting fake signatures into the record.[14] The earlier example involving Filament's Taps illustrates this point.

Cloud vs. Blockchain Models

In the cloud model, IoT devices are identified, authenticated, and connected through cloud servers,

Table 1. How blockchain can address Internet of Things (IoT) challenges.

Challenge	Explanation	Potential blockchain solution
Costs and capacity constraints	It is a challenge to handle exponential growth in IoT devices: by 2020, a network capacity at least 1,000 times the level of 2016 will be needed.	No need for a centralized entity: devices can communicate securely, exchange value with each other, and execute actions automatically through smart contracts.
Deficient architecture	Each block of IoT architecture acts as a bottleneck or point of failure and disrupts the entire network; vulnerability to distributed denial-of-service attacks, hacking, data theft, and remote hijacking also exists.	Secure messaging between devices: the validity of a device's identity is verified, and transactions are signed and verified cryptographically to ensure that only a message's originator could have sent it.
Cloud server downtime and unavailability of services	Cloud servers are sometimes down due to cyberattacks, software bugs, power, cooling, or other problems.	No single point of failure: records are on many computers and devices that hold identical information.
Susceptibility to manipulation	Information is likely to be manipulated and put to inappropriate uses.	Decentralized access and immutability: malicious actions can be detected and prevented. Devices are interlocked: if one device's blockchain updates are breached, the system rejects it.

where processing and storage are often carried out. Even if devices are a few feet apart, connections between them go through the Internet.[15]

First, IoT networks that have high costs are a concern in the centralized cloud model. Gartner estimated that in 2016, 5.5 million new IoT devices were connected every day.[16] It is estimated that by 2020, a network capacity that is at least 1,000 times the level of 2016 will be needed.[17] The amount of communication that needs to be handled will increase costs exponentially.

Second, even if economic and manufacturing challenges are addressed, each block of the IoT architecture could act as a bottleneck or point of failure that can disrupt the entire network.[18] For instance, IoT devices are vulnerable to DDoS attacks, hacking, data theft, and remote hijacking. Criminals might also hack the system and misuse data. If an IoT device connected to a server is breached, everyone connected to the server could be affected.

Consider smart water meters and associated risks. Twenty percent of California's residents have smart water meters, which collect data and send alerts on water leakage and usage to consumers' phones. Likewise, the Washington Suburban Sanitary Commission (WSSC) in Washington, DC, is planning to integrate IoT into its system. Water-usage data can tell criminals when residents are not home. Perpetrators can then burglarize homes when their residents are away.[19]

Third, the centralized cloud model is susceptible to manipulation. Collecting real-time data does not ensure that the information is put to good and appropriate use. Consider the water supply system example just discussed. If state officials or water service companies believe that the evidence might result in high costs or lawsuits, they can censor, edit, or delete data and analysis. They can also manipulate findings. For instance, consider the water crisis in the city of Flint, Michigan, which began in 2014. Flint authorities insisted for months that city water was safe to drink.[19] Citing official documents and findings of researchers who conducted extensive tests, a CNN article asserted that Michigan officials might have altered sample data to lower the city's water lead level.[20] It was reported that the Michigan Department of Environmental Quality and the city of Flint discarded two of the collected samples. A researcher said that the discarded samples had high lead levels. Including them in the analysis would have increased the level above 15 parts per billion (PPB). According to the US Environmental Protection Agency, water supply companies are required to alert the public and take action if lead concentrations exceed the "action level" of 15 PPB in drinking water (bit.ly/1qKMLVE).

Blockchain can eliminate many of the drawbacks described in Table 1. In blockchain, message exchanges between devices can be treated in a similar way as financial transactions in a bitcoin network. To exchange messages, devices rely on smart contracts. Blockchain cryptographically signs

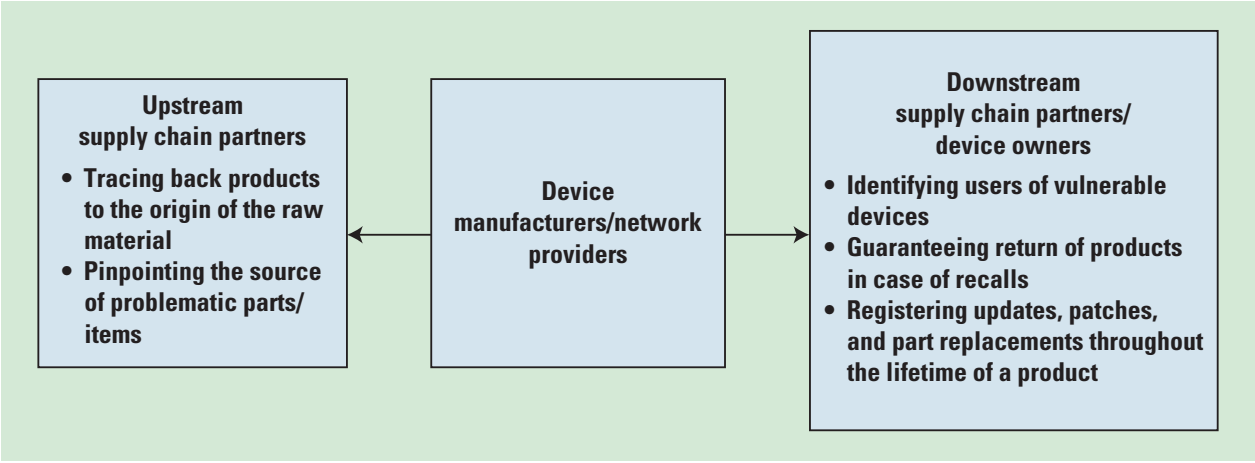


Figure 1. Blockchain's role in improving overall security in supply chain networks. With blockchain, it is possible to access immutable records for various aspects of transactions involving a product to understand key vulnerabilities in the upstream supply chain. This technology can also help strengthen downstream supply chain partners' and device owners' precautionary and defensive cybersecurity measures.

transactions and verifies those cryptographic signatures to ensure that only the message's originator could have sent it. This can eliminate the possibility of man-in-the-middle, replay, and other attacks.[6]

Blockchain's proponents have forcefully argued that this new technology can save us from "another Flint-like contamination crisis."[19] Projects such as the WSSC's integration of the IoT in supply systems can be upgraded with sensors such as near-infrared reflectance spectroscopy (NIRS) to include data on chemical levels. If such a system had been installed in Michigan, Flint's water service company could have found the lead contamination when it exceeded healthy levels. Blockchain can provide the "second layer of crisis prevention" in such cases.[20]

Ensuring Supply Chain Security

Blockchain can ensure supply chain security (see Figure 1). It also makes it possible to contain an IoT security breach in a targeted way after discovery of the breach. Blockchain can facilitate handling and dealing with crisis situations such as product recalls due to security vulnerabilities. Blockchain's public availability means that it is possible to trace back every product to the origin of the raw materials, and transactions can be linked to identify users of vulnerable IoT devices.

IoT-linked security crises, such as the cyberattacks on Dyn, could have been handled better if the supply chains had adopted blockchain. For instance, China-based Hangzhou Xiongmai Technologies, which makes Internet-connected cameras and accessories, recalled its products in the US that were vulnerable to the Mirai malware. However, it is difficult to determine the devices' owners. Blockchain is suitable for complex workflows. It can be used to register time, location, price, parties involved, and other relevant information when an item changes ownership. The technology can also track raw materials as they move through the supply chain, are transformed into circuit boards and electronic components, are integrated into products, and are sold to customers. Blockchain can also be used to register updates, patches, and part replacements applied to any product or device throughout its lifetime. It is easier to track progress in addressing vulnerabilities and send warnings and notifications to owners.[8]

Based on the evolving mechanisms and forces described here, a promising future seems likely for the use of blockchain in addressing IoT security. For instance, some of the key security challenges associated with the cloud can be addressed by using the decentralized, autonomous, and trusted capabilities of blockchain. Blockchain's decentralized and consensus-driven structures are likely to provide more secure approaches as the network size increases exponentially.

Blockchain enables the verification of the attributes it carries. Blockchain-based transactions are easily auditable. Due primarily to this and other features, blockchain

This article originally appeared in IT Professional, *vol. 19, no. 4, 2017.*

can play a key role in tracking the sources of insecurity in supply chains as well as in handling and dealing with crisis situations such as product recalls that occur after safety and security vulnerabilities are found. And as mentioned, blockchain-based identity and access management systems can address key IoT security challenges such as those associated with IP spoofing.

Acknowledgments

I thank Jeff Voas for numerous edits and suggestions on previous versions of this article. Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

References

1. K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, May 2016, pp. 2292–2303.
2. *Blockchain in Banking: A Measured Approach*, Cognizant Reports, 2016.
3. "3rd Cyberattack 'Has Been Resolved' After Hours of Major Outages: Company," NBC New York, 21 Oct. 2016; bit.ly/2eYZO46.
4. N. Perlroth, "Hackers Used New Weapons to Disrupt Major Websites Across US," *New York Times*, 21 Oct. 2016; nyti.ms/2eqxHtG.
5. E. Blumenthal and E. Weise, "Hacked Home Devices Caused Massive Internet Outage," *USA Today*, 21 Oct. 2016; usat.ly/2eB5RZA.
6. J. Coward, "Meet the Visionary Who Brought Blockchain to the Industrial IoT," *IOT World News*, 14 Dec. 2016; bit.ly/2s8la1w.
7. A. Kaul, "IBM Watson IoT and Its Integration with Blockchain," Tractica, 1 Aug. 2016; bit.ly/2rsOp2M.
8. B. Dickson, "Blockchain Could Help Fix IoT Security after DDoS Attack," *VentureBeat*, 29 Oct. 2016; bit.ly/2dXNaNO.
9. B. Dickson, "How Blockchain Can Change the Future of IoT," *VentureBeat*, 20 Nov. 2016; bit.ly/2qXZWXw.
10. S. Pajot-Phipps, "Energizing the Blockchain—A Canadian Perspective," *Bitcoin Magazine*, 26 Jan. 2017; bit.ly/2r7IIEc.
11. J. Brown, "Companies Forge Cooperative to Explore Blockchain-Based IoT Security," *CioDive*, 30 Jan. 2017; bit.ly/2quIMfv.
12. E. Young, "Tech Giants and Blockchain Startups Unite to Make IoT Apps More Secure," *The Cointelegraph*, 30 Jan. 2017; bit.ly/2kNtm7w.
13. C. Catallini, "How Blockchain Applications Will Move Beyond Finance," *Harvard Business Rev.*, 2 Mar. 2017; bit.ly/2m2ZIZQ.
14. S. Kumar, "Not Just for Cryptocash: How Blockchain Tech Could Help Secure IoT," IoT Agenda, 13 Feb. 2017; bit.ly/2m8H9Gr.
15. A. Banafa, "IoT and Blockchain Convergence: Benefits and Challenges," *IEEE Internet of Things* newsletter, Jan. 2017; bit.ly/2n1y8jq.
16. R. Van der Meulen, "Gartner Says 6.4 Billion Connected 'Things' Will Be in Use in 2016, Up 30 Percent From 2015," Gartner press release, 10 Nov. 2015; www.gartner.com/newsroom/id/3165317.
17. S. Waterman, "Industry to Government: Hands Off IoT Security," Fedscoop, 17 Nov. 2016; bit.ly/2g4oXYX.
18. A. Banafa, "A Secure Model of IoT with Blockchain," OpenMind, 21 Dec. 2016; bit.ly/2j2QUkH.
19. R. Hackett, "How Blockchains Could Save Us from Another Flint-Like Contamination Crisis," Venturebeat, 25 Feb. 2017; bit.ly/2mx11zp.
20. D. Debucquoy-Dodley, "Did Michigan Officials Hide the Truth about Lead in Flint?" CNN, 14 Jan. 2016; cnn.it/2r0aiF9.

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 nbkshetr@uncg.edu.*

Editor: **Daniel Zeng**, University of Arizona and Chinese Academy of Sciences, zengdaniel@gmail.com

Fully Autonomous Driving: Where Technology and Ethics Meet

Dieter Birnbacher, *University of Duesseldorf*
Wolfgang Birnbacher, *IBEO Automotive Systems GmbH*

Machines tend to be superior to humans in terms not only of strength and precision but also of reliability in controlling complex processes and the capacity to learn from mistakes. In motorized traffic, the advantages of fully autonomous driving are evident. Trucks driving in a densely packed convoy would greatly reduce gas consumption by reducing air resistance; autonomous cars that allow passengers to make good use of their travel time would greatly reduce time wasted. Above all, autonomous car traffic holds the promise of massive increases in safety. Especially in situations where presence of mind and reaction time are crucial, machines can be expected to perform better than humans. Instead of emotional and reflex-like reactions, a machine can analyze a situation in a split second and make decisions based on an algorithm established long in advance.

The prospect of automatized car traffic, however, confronts ethics, law, and politics with novel and far-reaching questions. Even if autonomous cars are constructed in a way that makes traffic safety the top priority, critical situations in which loss of life and limb are inevitable could still arise, requiring them to negotiate between two or more evils. How, for example, should the autonomous car react if it must choose between risking serious damage to one or more passers-by or risking serious damage to one or more of its passengers?

At this point, it is important to clearly distinguish the tasks of technology from ethics and the haunting challenges confronting each. On the technology side, an autonomous car's system of sensors and control algorithms must be able to substitute for a human driver and even surpass a driver in relevant capacities. To achieve this, normal signal processing techniques (sensor fusion, object classification, and so on) are not enough. The algorithms must incorporate large parts of a human driver's accumulated experience. Current sensor data must be integrated and processed together with the acquired understanding of complex contextual relations. The result of this process will be a statistical situation assessment with probabilities (risk and gain) for different possible reactions. Given the complexity and diversity of possible scenarios, the "right" reaction cannot be programmed in advance for every concrete situation, but has to be calculated according to a defined algorithm.

Despite the enormous complexity of these tasks, the challenges are purely technical. As with a driver's test, requirements for the quality of this situation assessment can be defined and tested with real and virtual test drives.

The ethical tasks to be mastered are no less challenging. Several hard questions must be answered:

- Who should have the authority to decide whether the preference rules and learning skills programmed into the system are acceptable?
- How much differentiation in ethical programming should be allowed to different stakeholders (producers, users, societies)?
- Who is responsible in case of damage?

Errata

In the article, "Using Process Mining to Model Multi-UAV Missions through the Experience," by Juan Jesús Roldán, Jaime del Cerro, and Antonio Barrientos (http://ieeexplore.ieee.org/document/8012327/), there are two errors in Table 1, specifically in column 3, rows 7 and 8. The IM-Inf should be "Model," whereas the IM-Inc should be "No model." We regret any confusion this error has caused.

Although decisions in dilemmatic situations have become popular as exercises in ethical judgment in university courses and even in high schools, they concern rare cases and are marginal in comparison with the more central questions confronting society. After all, programming a certain risk behavior into a machine not only has consequences in critical situations but also defines the driving style generally. How safe is safe enough? How safe is too safe? Excessive safety would paralyze road traffic and seriously hamper acceptance of autonomous vehicles. Giving leeway to risky driving styles would jeopardize the safety objectives. How egalitarian does an automatized driving system have to be? Is a manufacturer allowed to advertise with fast cars at the price of lowered safety for other road users?

Empirical studies suggest that a great majority of people prefer a decision algorithm that minimizes overall damage. At the same time, they seem to be prepared to accept reductions of their own safety as long as everyone accepts these same risks. This seems to imply that the level of acceptable risk can be calculated as a utilitarian optimum over all users, and that the risks produced by individuals is equal for all autonomous vehicles. It goes without saying that an egalitarian decision algorithm along these lines would lead to a radical shift of responsibility from the individual to the public. Neither the owner nor the passengers could be held responsible for the behavior of the vehicle any longer since risk preferences and conflict solving are determined in advance by societal consensus, leaving no room for individual intervention. The same holds for producers. Since the vehicle's decisions and reactions follow socially established norms, producers can no longer be held responsible for damages that occur as a consequence of these norms.

This means that society, represented by its respective legislators, must establish the general principles of the decision algorithm and assign weights to the different kinds of evils in a socially acceptable way. More than any other area, technology, once put into use, is inseparable from ethics.

Even if a social consensus might be achieved on a broadly utilitarian ethical framework, there is plenty of room for controversy. One question is how to deal with questions of fairness—for example, between vehicles that are more and less vulnerable in crashes. A decision algorithm programmed to minimize harm would choose collision with a smaller vehicle in an unavoidable swerve maneuver over collision with an "armored" SUV because the latter would involve greater risk for its passengers. Owners of smaller cars who cannot afford bigger ones would likely therefore feel discriminated against. There might be controversy also about the weight assigned to different goods in potential conflict situations, such as the bodily integrity of humans and animals.

Acceptance of autonomous driving will depend on how far a consensus on these norms can be found, first among experts, then in society at large. One ethical condition, however, should be crucial: in no case should the ethical algorithms be put in practice as nontransparent black boxes. The built-in norms should, as far as possible, be understood and commonly shared. ■

Dieter Birnbacher is a professor of philosophy at the University of Duesseldorf. His research interests include ethics and applied ethics. Birnbacher has an honorary doctorship in philosophy from the University of Muenster, Germany. He is a member of Leopoldina, National Academy of Sciences. Contact him at dieter.birnbacher@uni-duesseldorf.de.

Wolfgang Birnbacher is an FPGA system designer at IBEO Automotive Systems GmbH. His research interests include sensor algorithms and signal processing. Birnbacher has an MASc in microelectronic systems from HAW Hamburg. He is a member of IEEE. Contact him at wolfgang.birnbacher@ibeo-as.com.

This article originally appeared in IEEE Intelligent Systems, *vol. 32, no. 5, 2017.*

by Charles Day

Computer-Aided Fashion

My wife, Jan, surprised me last year while we were watching the Super Bowl. During the halftime show, which was unmemorably headlined by Coldplay, she asked if I would wrap her torso in duct tape.

Jan, it turned out, wanted to create a dressmaker's dummy. Following a recipe she had found online, she donned a sacrificial form-fitting T-shirt. My task was to apply strips of duct tape to the T-shirt until it was completely covered by at least three layers of snugly wrapped tape. In the final step, I used scissors to cut open the back so that Jan could remove the semi-rigid carapace that would serve as the dummy.

Making, say, a blouse from a pattern is an analog process. The first step, called grading, is to resize the pattern's pieces for a wearer's measurements. Those pieces, which take the form of pieces of paper, are then used as templates for cutting out the fabric. At that point, the dressmaker could sew all the pieces together and be done, bar pressing the garment. But if she wants a better fit, she could instead sew the pieces loosely together, drape the garment over the dummy, and then trim the pieces and adjust where the permanent stitching should go.

How might computation help this process? Having asked myself that question, I searched online for dressmaking software. At the low end, I found programs that tackled the problem of grading a pattern based on the wearer's measurements. At the high end were suites of CAD software that enabled the user to design a garment and then create a pattern. The fanciest software could even specify the least wasteful way of cutting pattern pieces from a bolt of cloth.

Two things struck me about the dressmaking software. First, the choice was modest: few had Mac versions and most hadn't been updated for years. Second, given what computer-aided fashion could conceivably be, the software packages seemed unambitious in their capabilities.

To see why, consider a somewhat challenging garment to make from scratch: a pencil skirt. Although patterns for a pencil skirt typically contain just a few pieces and have a small number of seams, the garment requires judiciously positioned darts to achieve a flatteringly snug fit. Darts are a dressmaker's way of dealing with the fact that the human body is curved whereas fabric is flat. Cartographers meet a similar challenge by projecting a map of Earth's globe onto cones, cylinders, and other solids whose so-called developable surfaces can be unrolled into a sheet without shrinking, stretching, or tearing.

The surface of a dressmaker's dummy is manifestly not developable, but you could imagine software that could virtually and arbitrarily transform it into a manageable set of developable surfaces. Using the software, a dressmaker could then design a pencil skirt made without darts but with a quirkily original set of pieces.

Unfortunately, software companies aren't in business to indulge the sartorial fancies of columnists. They're in business to make money. In 2010, clothing accounted for just 3 percent of annual disposable income in the US, down from 9 percent in 1950. Given how cheap clothing has become, making clothes at home isn't an economic necessity but a recreational pastime—one that's all the more satisfying for being analog, not digital. ■

This article originally appeared in Computing in Science & Engineering, *vol. 19, no. 3, 2017.*

Charles Day is *Physics Today*'s editor in chief. The views in this column are his own and not necessarily those of either *Physics Today* or its publisher, the American Institute of Physics.

 Published by the IEEE Computer Society

Careers in Software Engineering

For this *ComputingEdge* issue, we interviewed Murray Cantor—cofounder and chief technology officer of Aptage, an agile-software-development risk management consultancy—about career opportunities in software engineering. He has developed cutting-edge ideas in software and systems development for more than 35 years. In addition to writing many articles, he is the author of two books: *Object-Oriented Project Management with UML* and *Software Leadership: A Guide to Successful Software Development*. He coauthored the article "Steering Software Development Workflow: Lessons from the Internet" from *IEEE Software*'s September/October 2016 issue.

ComputingEdge: In the field of software engineering, what would you tell college students to give them an advantage over the competition?

Cantor: They should learn programming and implementation technologies for machine learning and AI, such as Nvidia's CUDA parallel-computing platform and API model, and the TensorFlow open source software library for machine intelligence.

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

Cantor: Applicants should avoid dead-end jobs using obsolete technology.

ComputingEdge: How can new hires make the strongest immediate impression in a new position?

Cantor: Show your manager a project you've done on your own initiative—not a class assignment—ideally something involving AI. In addition, be ready to provide good examples of team and leadership skills. Development is a team sport.

ComputingEdge: Name one critical mistake for young graduates to avoid when starting their careers.

2469-7087/17/$33.00 © 2017 IEEE Published by the IEEE Computer Society December 2017

Cantor: Avoid getting complacent with your skills. Software development careers entail lifelong learning. What matters is your ability to contribute to your team's success.

ComputingEdge: Do you have any learning experiences you could share that could benefit those just starting their software-engineering careers?

Cantor: When I was about 30, object-oriented programming was the latest thing. The conventional thinking among the younger programmers was that I was already too old to get it. I quickly learned the Booch method and C++, and proved them wrong. You will age and will need to keep up.

ComputingEdge's Lori Cameron interviewed Cantor 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 Cantor at mcantor@murraycantor.com.

IEEE computer society

SkillChoice™ Complete

Now with expanded libraries and an upgraded platform!

Valued at **$3,300!**

3,000+ online courses

MENTORSHIP

Practice Exams

28,000+ BOOKS

6,000+ VIDEOS

15,000+ *Books24x7 titles*

OVER 20x as many resources as before

One membership.
Unlimited knowledge.

Did you know IEEE Computer Society membership comes with access to a high-quality, interactive suite of professional development resources, available 24/7?

Powered by Skillsoft, the SkillChoice™ Complete library contains more than $3,000 worth of industry-leading online courses, books, videos, mentoring tools and exam prep. Best of all, you get it for the one low price of your Preferred Plus, Training & Development, or Student membership package. There's something for everyone, from beginners to advanced IT professionals to business leaders and managers.

The IT industry is constantly evolving. Don't be left behind. Join the IEEE Computer Society today, and gain access to the tools you need to stay on top of the latest trends and standards.

Learn more at **www.computer.org/join**.

ACCESS TO SKILLSOFT IS AVAILABLE WITH