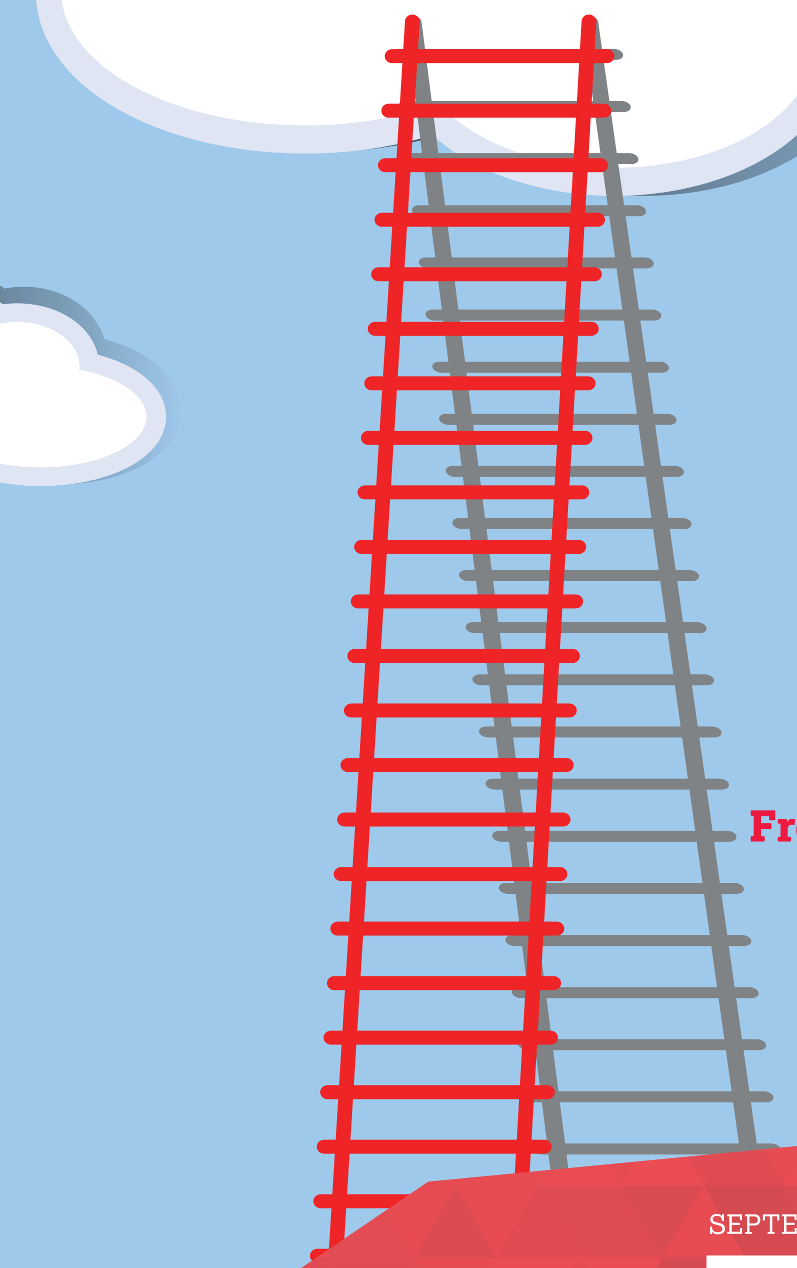
COMPUTING edge



Cloud Computing

Also in this issue:

> Multimedia Search: From Relevance to Usefulness

> Enhancing Engineering Productivity

SEPTEMBER 2015

www.computer.org

IEEE

IEEE computer society

COMPUTING edge

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

Printed with inks containing soy and/or vegetable oils

IEEE

IEEE computer society

BPA WORLDWIDE

Circulation: ComputingEdge 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 undelivered copies and address changes to IEEE Membership Processing Dept., 445 Hoes Lane, Piscataway, NJ 08855. Application to Mail at Periodicals Postage Prices is pending at New York, New York, and at additional mailing offices. Canadian GST #125634188. Canada Post Corporation (Canadian distribution) publications mail agreement number 40013885. Return undeliverable Canadian addresses to PO Box 122, Niagara Falls, ON L2E 6S8 Canada. Printed in USA.

SEPTEMBER 2015 • VOLUME 1, NUMBER 9

COMPUTING edge

Departments

55

Cognitive Assistance in the Workplace

Multivariate Network Exploration and Presentations

Shixia Liu, Tsinghua University

Leila De Floriani, University of Genova

This installment highlighting the work published in IEEE Computer Society journals comes from IEEE Transactions on Visualization and Computer Graphics.

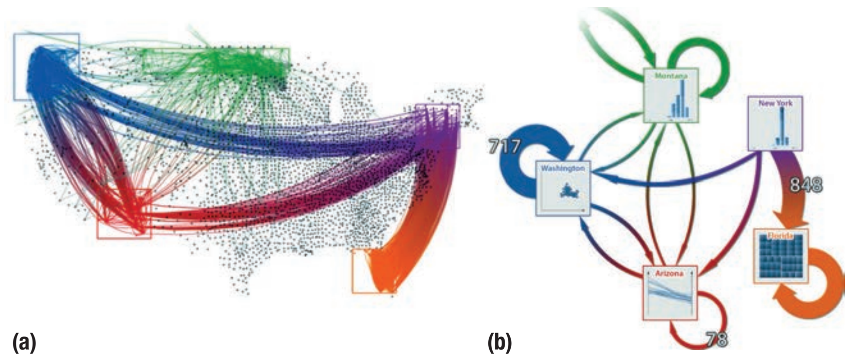


Figure 1. Large multivariate network exploration using selections of interest. (a) Detailed view. (b) High-level, infographic-style overview.

In many big data applications, it's important to explore and understand multivariate networks, whose nodes and edges contain multidimensional attributes. A core challenge is connecting big data with people; that is, presenting—in an understandable and manageable way—the structural and multivariate aspects of data to humans so they can explore both aspects simultaneously to gain insights on the data. In "Multivariate Network Exploration and Presentation: From Detail to Overview via Selections and Aggregations" (*IEEE Trans. Visualization and Computer Graphics*, vol. 20, no. 12, 2014, pp. 2310–2319), Stef van den Elzen and Jarke van Wijk propose a novel solution to this challenge.

When analyzing multivariate networks, users often want to understand the relationship between the network structure and its elements' properties to answer questions like whether communication patterns differ between males and females, or whether people migrate to places with similar characteristics.

To answer such questions, the authors present an approach based on an interactive exploration of the data, from detailed views to high-level overviews via selections and aggregations. For example, users can interactively select a set of nodes. The results are then shown as a detailed view of the nodes and their connecting edges and as a schematic diagram in a high-level overview (see Figure 1), which facilitates exploration. A demonstration video is available at www.win.tue.nl/~selzen/Elzen-Wijk-InfoVis-2014.mp4.

Building on an important trend in visualization, this work combines standard, simple, and familiar metaphors to explore complex data. As shown in Figure 1, the visualization components are standard and familiar: a scatterplot for showing details, an infographic-style overview, and a set of chart visualizations. However, the proposed combination is novel and effective for both data presentation and exploration.

Van den Elzen and van Wijk focus on helping users interact with and explore data by combining standard presentations, rather than displaying large, complex data sets in a single visualization with complex visual encoding. In doing so, the authors highlight an important design mantra, relevant for big data research, that goes beyond visualization.

SHIXIA LIU is an associate professor in the School of Software at Tsinghua University, China. Contact her at shixia@tsinghua.edu.cn.

LEILA DE FLORIANI is a professor of computer science at the University of Genova, Italy, and an adjunct professor at the University of Maryland, College Park. Contact her at ldf4tvcg@umiacs.umd.edu.

See **www.computer.org/computer-multimedia** for multimedia content related to this article.

Magazine Roundup

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

Computer

Irregular applications require special strategies to best handle unpredictable memory-access patterns, data-dependent control flow, and fine-grained data transfers. This is the topic of *Computer*'s August 2015 special issue.

IEEE Software

Software processes and design quality are interrelated, so developers must consider their impact on each other to ensure a high-quality result, according to the authors of "Software Process versus Design Quality: Tug of War?" from *IEEE Software*'s July/August 2015 issue.

IEEE Internet Computing

Potential users have many hopes for the Internet of Things (IoT). However, as the IoT ecosystem evolves, **dependencies on other technologies** could affect the design and resilience of the infrastructure, notes Internet pioneer Vinton Cerf in "Dependencies," which appears in *IEEE Internet Computing*'s July/August 2015 issue.

Computing in Science & Engineering

Testing scientific software can be thorny. In scientific programming, **computational approximations**—designed to convert an intractable problem into a tractable one—are closely related to software testing. "The Approximation Tower in Computational Science: Why Testing Scientific Software Is Difficult," from *CiSE*'s July/August 2015 issue, uses a solar-system simulation to illustrate these approximations and their role in testing scientific software.

IEEE Security & Privacy

Broad, multidisciplinary perspectives are becoming increasingly important as technology becomes more pervasive. It's critical to think about security and privacy approaches broadly to better understand the context in which they are perceived and used. This is the basis for *IEEE*

S&P's July/August 2015 special issue on multidisciplinary security. Several articles describe how applications of different techniques can lead to important security and privacy insights.

IEEE Cloud Computing

Researchers envision using cloud computing with mobile devices to overcome smartphone applications' ever-increasing computational and energy demands. However, this requires specialized, context-aware development models for creating cloud applications able to make context-aware computation-offloading decisions. "Context-Aware Mobile Cloud Computing and Its Challenges," from *IEEE Cloud Computing*'s May/June 2015 issue, provides an overview of how these issues affect **mobile cloud computing**.

IEEE Computer Graphics and Applications

The development of big data visualization and visual-analytics tools has focused on information that is important to science, business, and government. However, big data also affects individuals' personal and professional lives. *IEEE CG&A*'s July/August 2015 special issue discusses how **personal visualization and personal visual analytics** offer substantial opportunities to help people learn about themselves and their communities.

IEEE Intelligent Systems

Ambient assisted living (AAL) uses information and communication technologies in personal and work environments to help elderly people stay active, socially connected, and independent. AAL research covers many topics, and one of the most important is human-activity recognition and behavior understanding. This is addressed in *IEEE Intelligent Systems*' July/August 2015 special issue on AAL.

IEEE MultiMedia

IEEE MultiMedia's July–September 2015 issue on **social multimedia and storytelling** touches on many significant aspects of multimedia retrieval, including content analysis and understanding. The issue looks at challenging research problems in this emerging area, as well as topics linked to important commercial and creative applications in sectors such as the media, entertainment, arts and culture, sports, and music.

IEEE Annals of the History of Computing

Risk management is widely seen as the basis for cybersecurity in contemporary organizations, but some practitioners continue to dispute its value. "Measuring Risk: Computer Security Metrics, Automation, and Learning," in *IEEE Annals*' April–June 2015 issue, analyzes the debate over US **computer security risk management in the 1970s and 1980s**.

IEEE Pervasive Computing

The history of wearable computing dates to the 1980s, when it was employed largely to augment users' senses and capture their everyday experiences. Recent developments have fundamentally changed the digital augmentation of reality. *IEEE Pervasive Computing*'s July–September 2015 special issue on **digitally enhanced reality** explores this topic.

IT Professional

IT Pro's July/August 2015 special issue on **data analytics** focuses on four areas related to knowledge development and utilization: the field of education, optimizing knowledge sharing in communities, gaining and applying business insights, and the state of the technology in China.

IEEE Micro

Datacenter computing is trending toward large, shared hardware platforms, which poses two architectural challenges: sharing fairly and sharing multiple resources. Drawing on economic game theory, the authors of "Sharing Incentives and Fair Division for Multiprocessors," from *IEEE Micro*'s May/June 2015 issue, rethink **fairness in computer architecture** and propose a way to find fair allocations while maintaining system effectiveness.

Computing Now

The Computing Now website (http://computingnow.computer.org) features **up-to-the-minute computing news** and blogs, along with articles ranging from peer-reviewed research to opinion pieces by industry leaders.

EDITOR'S NOTE

Behind Every Cloud

In its relatively recent history, cloud computing has had a powerful effect on technology developers, providers, and users.

Individuals and companies no longer have to host expensive computing infrastructures but instead can economically access services and resources whenever and in whatever amount desired. This has also created many opportunities for new cloud-computing businesses.

This issue of *ComputingEdge* highlights recent developments in cloud computing. *Computer* magazine's "Software Modernization Revisited: Challenges and Prospects" discusses important factors to consider when migrating software to the cloud.

"The Modern Cloud-Based Platform," from *IEEE Software*, is the topic of an interview about architecture, development, and operations that make the most of cloud-based offerings.

In *IEEE Security & Privacy*'s "Confidentiality in the Cloud," the authors say that security problems have reduced the cloud's appeal and that the technology requires fundamental changes to provide the necessary privacy.

Cloud storage's rapid growth has made it challenging for storage architects to meet customers' diverse performance and reliability requirements while controlling costs. Erasure-coded and software-defined storage could address these challenges, according to *IEEE Internet Computing*'s "The Growing Pains of Cloud Storage."

Cloud standards have been limited by the rapidly changing nature of the technology. *IEEE Cloud Computing*'s "Socioeconomics of Cloud Standards" discusses factors that could encourage their development.

To choose the most appropriate cloud-computing model, an organization must analyze its IT infrastructure, usage, and needs, notes *IEEE Software*'s "Infrastructure as a Service and Cloud Technologies."

Now that the cloud-services market is maturing, it's time for the cloud-computing pricing strategies and market structure to advance beyond the elementary, explains "What's the Best Way to Purchase Cloud Services?" from *IEEE Cloud Computing*.

Articles on topics other than cloud computing in this issue include:

- *IEEE Intelligent Systems*' "Human Beyond the Machine: Challenges and Opportunities of Microtask Crowdsourcing";
- *IEEE Computer Graphics and Application*'s "High Tech @ Home," which features recent advances in home-oriented computer graphics technology;
- *IEEE Internet Computing*'s "Lest We Forget," in which Internet pioneer Vinton Cerf discusses how digital information can best be preserved for future access;
- *IEEE MultiMedia*'s "Multimedia Search: From Relevance to Usefulness";
- *IEEE Pervasive Computing*'s "Cognitive Assistance in the Workplace";
- *Computing in Science & Engineering*'s "Enhancing Engineering Productivity"; and
- *IT Professional*'s "The Need for New Business–Technology Relationships.".

Editor: **Robert Blumen**
Symphony Commerce
robert@robertblumen.com

The Modern Cloud-Based Platform

Stefan Tilkov

THIS MONTH'S EXCERPT from Software Engineering Radio (www.se-radio.net) features the third in a recent series of podcasts touching microservices, beginning with *210: Stefan Tilkov on Microservices* and *213: James Lewis on Microservices*. Docker was the subject of the fourth podcast, episode 217, in which Charles Anderson tracked down that project's founder for an in-depth discussion.

Netflix established itself as a company by disrupting the video rental industry with monthly pricing, a deep back catalog, and machine-learning-based recommendations. But it's barely in the DVD business now. The second Netflix revolution has been an aggressive move into streaming content from the Amazon cloud. Netflix and its US customers are one of the heaviest users of the US Internet, by some estimates accounting for over one-third of all traffic during peak movie-watching hours. Its movie-streaming business has been the source of technological innovations as it has innovated to meet consumers' ever-rising standards.

Much of what Netflix has learned is now public information through the Netflix open-source stack and the tireless efforts of Adrian Cockroft (formerly of Netflix), one of the major cloud architects during his years there. Cockroft has been a regular speaker at tech conferences, many of which are online.

In SE Radio episode 216, Cockroft and our newest host, Stefan Tilkov, converse about Netflix's move to the cloud, development speed as the critical competitive factor, how the monolith gave way to microservices, microservices at scale, microservices and DevOps, the Netflix service discovery infrastructure, distributed debugging in a deep microservices stack, geographic redundancy on the Amazon cloud, being always on, availability over consistency, the strategic plan behind open source, how open source helps with hiring in a competitive market, and what Adrian is doing post-Netflix in the venture-capital field.

We would enjoy hearing from readers of this column and listeners to the podcast. We accept incoming emails at se-radio@computer.org and tweets and direct messages to @seradio. You can also visit our Facebook page, Google+ group, and LinkedIn group. To hear this interview in its entirety, visit www.se-radio.net. —Robert Blumen

It's hard to read any sort of article these days that doesn't somehow mention the cloud or cloud computing. Despite that, how do you define those things? What is the cloud, and what does cloud computing mean to you?
The biggest change is when someone working at a company thinks, "I need some machines to do something." You have to file a ticket and wait for somebody else to get around to sorting out that ticket. In some big companies, it

 Published by the IEEE Computer Society

takes a month or two to get just a single machine. The real thing that makes it cloud computing is self-service. You make an API call, and a few minutes later a bunch of machines turn up. That is the most fundamental difference. It speeds up the whole procurement cycle. It makes everything much more dynamic.

You can use the cloud as a faster way to do the things you used to do in datacenters. But the really interesting things come when you start realizing what happens when you use it in a much more dynamic way by using machines as ephemeral resources. You can turn them on, turn them off, use a machine for a couple of hours, and then give it back. That is the essence of cloud. It comes down to putting self-service tools in the hands of the developers to do things themselves, rather than making "what operations used to do" into an API.

Developers don't care whether it's a public cloud like AWS [Amazon Web Services] or a private cloud inside the company, as long there's enough capacity for whatever you need to do. It's just there.

How did Netflix end up moving to the cloud?

Netflix started off as a DVD shipping company. It wasn't even seen as a very big technology company at the time. When I joined, its personalization algorithms were considered its primary interesting technology, not its scale.

Netflix had around 6,000,000 customers when I joined in 2007. Typically, every weekend the customers would visit the website, decide what DVDs they wanted to have shipped in the next week, and prioritize by shuffling their queue. Every time they sent a disc back, we would send them another one. The interaction with Netflix was sending a disc in through the postal service. That required a few tens of Web servers, a few back-end machines, and a big database running on a monolithic centralized app.

That year, we launched streaming with a very small catalog, but it started to take off quite quickly. When you interact with a streaming service, every click goes to the website. You browse around the website. When you decide that you want to watch something, you click Play. Then traffic goes back and forth figuring out what to do: finding the machine, finding the movie, doing authorization, giving you a security key to decode that movie, and then logging the activity so that we can make sure that there's good quality of service (rebuffers, calculating the speed to run at, and determining which content delivery network to use).

There are enormously many transactions to the back end. Streaming generates around a thousand times more Web requests than the DVD service. That was fine when we were just starting out with a small number of machines, a small number of movies, and not many customers using it. But the usage rate started to increase very rapidly because there was nothing stopping customers from watching a lot of movies. They no longer had to send a DVD back and wait for another one. So, we started getting a lot of people binge-watching.

The number of interactions people had with Netflix, the number of things they watched, and the number of interactions with the website per view all went up by orders of magnitude. Our datacenter consisted of a couple of small machines in the corner that were put in to initially launch streaming. And they were running out of capacity incredibly quickly. In 2008, there was a big outage when the central monolithic app broke due to a storage problem (a corruption in the storage area network corrupted Oracle). It was a big mess.

As a result, we decided we were not very good at running stuff in the datacenter. We started thinking about how to scale for this incredible future workload, where we didn't know how fast it would grow. And that really comes down to the core of why this is interesting: because we could not predict how much capacity we would need.

In 2009, we moved some of the back-end batch workloads like encoding movies to the cloud. In 2010, we moved the front-end website to the cloud. In 2011, we moved the database back end so that the master copies of all the data were in the cloud, and in 2012, we started open-sourcing the tooling we'd built to do that. Those are the main history points.

You've hinted that software architecture changes when you move to the cloud. What changes?

You can use cloud as nothing more than a faster way to do datacenter stuff, but that misses most of the benefit. The real benefit comes when you start doing things that you couldn't have done in your datacenter. You can trivially do hardware experiments that last a few days on a huge scale, scattered all over the world, something that you wouldn't even think of doing if you had to tell your ops guys, "Hey, I need a liberally distributed database with a hundred nodes in it and a couple hundred terabytes of solid state disc. And I'd like it this afternoon, in six different datacenters. And whatever."

We did this and we did it without asking permission. It took about

SOFTWARE ENGINEERING RADIO

Visit www.se-radio.net to listen to these and other insightful hour-long podcasts.

RECENT EPISODES

- 217—Charles Anderson talks with James Turnbull, the creator of Docker, a popular lightweight Linux deployment tool. Somewhere between a process and a virtual machine, Docker is emerging as a container for isolating microservices.
- 218—Robert Blumen discusses the Command-Query-Responsibility Segregation (CQRS) architectural pattern with Udi Dahan, one of the pattern's cocreators. CQRS formally separates a distributed system into a write master and one or more read models.
- 219—Jeff Meyerson interviews Apache Kafka committer Jun Rao on the high-throughput distributed event bus that combines features of messaging and publish–subscribe.

UPCOMING EPISODES

- 220—Robert Blumen sits down in person with Jon Gifford for a conversation about logging, APIs, log record formats, and how search engines are transforming the collection and interpretation of program messages.
- 221—Johannes Thönes and chief guru Jez Humble converse on the origins of continuous delivery (CD) in the lean movement, how to build a CD culture, and how to introduce CD in regulated environments.
- 222—Apache Storm Founder Nathan Marz chats with Jeff Meyerson about stream processing, "real-time Hadoop," the lambda architecture, and thinking about streaming in terms of spouts and bolts.

20 minutes to create. We built a very write-intensive globally distributed database to see what would happen. The decision to do it was made while we were walking out of a meeting. The guy who did it wandered by somebody else's cube and asked that person to set it up. That afternoon it was created. We put 18 Tbytes of data from backup on it. And then we hammered the thing as hard as we could with all kinds of error and failure injections to make sure it worked well. A few days later we removed it.

You couldn't do that if you had real infrastructure because it would take too long to get approval. In a datacenter, you would need a multimillion-dollar machine. I didn't know in advance what it would cost, but it worked out to a few hundred dollars per hour. The value we got out of it was much greater: at a meeting the following week, we said, "We just proved this works."

At the time, there was an internal argument going on about how we would build distributed systems and whether we could rely on high bandwidth in a global cloud. When you go to a meeting with working code or benchmarks, you win arguments. That short-circuited an enormous amount of what would have been debate and justification.

Can you explain the Chaos Monkey?

I will explain the principle, and then it will be obvious that it makes sense. There is an analogy of cattle versus pets. If you know your machines in production by name, and if one goes down everyone gets upset, then that's a pet. You have to take it to the vet if it gets sick. The other kinds of machines are cattle. When you have a herd of cattle in the field, they produce so many gallons of milk. If a few of them die, you get slightly less milk that day, and you buy some more cattle.

The principle that Netflix adopted was that everything in production is a herd of cattle. There are no pets, no individual machines that if one went down anybody would care about. Everything is on an autoscaler, even a single machine.

Once you have established that principle, then you have to test that compliance by killing individual machines chosen at random. That is what the Chaos Monkey does. It picks a random time to stop some machines. The autoscaler should automatically replace it. If somebody snuck an individual machine into production, the Chaos Monkey killed it, and they got upset, well, they should not have done that, right? It forces the developers to think in the new way. Everything you launch is on an autoscaler even if there is only one machine in the group. It must be a stateless, disposable machine that can be restarted.

They took it to the next level

with the data layer, which is a triple-replicated Cassandra back end. The Chaos Monkey kills those as well—including the discs that are inside the instances. You might lose a few hundred gigabytes of data when you delete the instance. It's not attached storage: the disks are inside the instance. But it's replaced as the data is resynchronized from the other two copies, proving that you can build an ephemeral data layer as well.

This is a different way of thinking from a sort of datacenter mentality where machines should always stay up. You would use perfect machines if you could. Instead, you create herds of machines that are extremely resilient because you can lose large numbers of them and everything still works.

Let's talk about the CAP (consistency, availability, partition tolerance) theorem. Which part of it do you apply? Which side of the triangle do you lean to?

You have to decide whether consistency or availability is most important to you. The basic principle of Netflix is that no partition or failure should take out the service. We lean very heavily to the availability side when things are partitioned, which means that if you slice Netflix up and drop all the networks between all the different parts of the system, the system continues to work. The isolated parts will just carry on working. And they'll gradually become inconsistent with the rest of the system. When you reconnect, "last writer wins" takes over. Cassandra kicks in. Whoever wrote a piece of data last ends up overwriting whatever was written in the meantime.

The system gradually gets back to being consistent, although you may lose a few updates if you modified something that was modified somewhere else. Generally it doesn't, and anyway, it's better to deal with inconsistency than to be down. For a service like Netflix, where 50 million people are trying to watch movies around the world, they have an expectation that when they turn on a TV set, it should just work. There should never be a message saying, "We're down right now."

It's hard to tell a three-year old that they can't watch their dinosaur cartoons because Netflix has failed.

Yeah, tell me about it.

STEFAN TILKOV is cofounder and principal consultant at innoQ, a technology consulting company with offices in Germany and Switzerland. Contact him at stefan.tilkov@innoq.com.

This article originally appeared in IEEE Software*, vol. 32, no. 2, 2015.*

See www.computer.org/software-multimedia for multimedia content related to this article.

What's the Best Way to Purchase Cloud Services?

James Mitchell
Strategic Blue

IT'S NOT NEWS THAT UPTAKE OF PUBLIC CLOUD INFRASTRUCTURE AS A SERVICE (IAAS) HAS BEEN GROWING. It's less obvious how IT purchasing strategies should shift to best leverage the availability of different pricing options for these utility IT services, such as on-demand, reserved or committed, and whether to buy direct or utilize new distribution channels. At Strategic Blue, which acts as a financial cloud broker—that is, a financial intermediary between cloud service providers (CSPs) and customers—we've had a front row seat to the changes rocking the industry, and developed some insights into the best strategies for customers to adopt. The cloud, which launched with the simplest pay-as-you-go or on-demand tariff, was supposed to unleash a new era of simplicity in procuring IT services. Somewhat surprisingly, however, pricing models designed to entice production workloads into the cloud have created their own complexities, and are thus transforming where value is created and how it's captured.

EDITOR
JOE WEINMAN
joeweinman@gmail.com

The Shifting Channel

A fundamental shift is underway from tangible products to utility services, and from ownership to access, affecting all aspects of the IT distribution channel. The last several decades have seen the development of a complex supply chain stretching from manufacturers of components, such as CPUs and disk drives, to assemblers of hardware encompassing compute, storage and networking, and then on to distributors and value-added resellers of these tangible hardware products. Often software is bundled, either by an assembler, distributor, or reseller. This bundling might include installation and even some degree of configuration, and as a result, the customer often outsources the acquisition of software licenses to a company upstream in the supply chain.

The change from physical supply of tangible IT systems to the provision of on-demand access to cloud services under a utility (pay-per-use) delivery model transforms that entire supply chain. Even large distributors—with their high volume, typically low margin, and multibillion-dollar revenue streams—are under increasing pressure to adapt their business models to address this profound change. Similar pressures are being felt by IT resellers of all sizes traditionally served by such distributors. The past two years in particular have seen even the largest vertically integrated IT companies respond directly to the emergence of cloud services by developing new services or making acquisitions.

However, there are significant differences between selling a tangible systems product and selling a utility service. If the resale model for utility services is poorly structured, a host of unintended, adverse consequences can occur, the most damaging of which is a lack of pricing consistency.

 Published by the IEEE Computer Society

The Need for Pricing Consistency

Customers such as CIOs, IT executives, and their procurement colleagues tend to assume that cutting out a middleman saves money. Although in many cases this might be true, it's also true that a value-added reseller should, after all, be adding value, not just cost. Skeptical customers want middlemen such as resellers to justify, and ideally quantify, the costs and benefits of their involvement. This isn't an easy task without intimate knowledge of both the customer and the distributor. The most immediate test that customers will apply is to compare the reseller's price with the price for purchasing a service directly from the service provider. This is where one can walk into a fog of cloud pricing complexity, due, for example, to differences in contractual terms; the bundling together of IT resources, support, and consultancy; and varying technical performance between CSPs' service offerings.

Moreover, some CSPs, such as Channel Cloud or Faction (formerly known as Peak Colo), don't sell directly to end customers, preferring to enable partners to offer white label solutions, so the comparison can't always be accomplished. In such cases, this leaves only comparisons among resellers as a basis for a purchasing decision. This model is "cleanest" in that it's closest to evaluating comparable offerings.

Most CSPs do sell direct to end users. However, many of them are willing to bypass list prices and negotiate special deals for special customers. However, customers talk not just to vendors or service providers, but to each other. Therefore, as word of special deals gets around, every customer wants one, lengthening the sales cycle and driving up the cost of each sale, a cost ultimately borne by the customer.

These undisciplined CSP sales practices can then have a cascading effect on the CSP's resellers. A potential customer will attempt to circumvent the reseller by various means in order to determine the direct pricing from the service provider, and thus establish just how special a "special price" really is. A handful of customers can thus expose the nature and range of variability in special pricing. It further puts a reseller's credibility at risk if a potential customer can establish that he or she is being quoted a special price that isn't as deeply discounted as that offered to another customer. This is a problem not just for resellers, but also for providers, and surprisingly, even customers.

> The cloud, which launched with the simplest pay-as-you-go or on-demand tariff, was supposed to unleash a new era of simplicity in procuring IT services.

Pricing variability and its disclosure to customers undermine the CSP's ability to sell through the channel, defeating the whole point of setting up a resale channel in the first place. Moreover, customers are impacted through reduced access to cloud services, longer procurement cycles, and higher costs due to additional complexity.

A solution to this problem is for the CSP to establish wholesale pricing. The CSP establishes a uniform price at which the cloud services are sold to both its internal sales team and its resale channels. The critical success factor for such an approach is that the CSP must see to it that this pricing can't be circumvented, even by its own direct sales team. To minimize complexity and room for confusion, CSPs might establish a very limited set of pricing structures, such as pay-as-you-go and prepay. If the sales team or the resale channel needs a pricing structure other than that, the CSP might employ a third-party financing/risk-management solution. (Disclosure: this is one of Strategic Blue's offerings.)

How Can a Cloud Buyer Procure at the Best Price?

The classic approach to finding the best price is to ask a number of vendors to respond to a request for proposal (RFP) or a request for quotation (RFQ). When IT was synonymous with hardware and software products, this worked fairly well, as vendors were geared up for this approach and had a team of sales and sales support personnel focused on responding to RFPs and RFQs. However, the traditional model doesn't translate directly, because experimentation is cheap and self-service is the order of the day. Cloud services are often provided initially through a limited free trial, after which a customer can do a larger scale paid trial for a trivial cost, typically paid by credit card. CSPs try to stay lean and cost competitive by automating as much as they can, including the sales process, so they have far fewer salespeople ready and waiting for a would-be buyer's RFQ to be released.

Thus, if an RFP calls for a number of non-industry-standard terms requiring special sign-off, unless the customer's potential cloud spend is enormous, that customer might not attract any responses at all, or only responses from those cloud providers who have invested in people to respond to tenders, instead of investing in people to build a great cloud service. This could be where resellers have an advantage—they're geared up to respond to tenders. Of course, to enable this option, the tender must not preclude resellers from bidding.

Suppose, however, that the RFP was standard enough to attract a range of bids from the best providers. All the bidders will have quoted pricing using their own pricing structures and even their own proprietary units of measure. For example, one provider might offer virtual servers with fixed combinations of CPU, RAM, and storage; another allows tailoring of these resources. Some will show pricing based on prepay, others will offer 30-day payment terms. Some will bundle resources together; others will price everything in microscopic detail. To make this even more complex, the ultimate cost depends not just on the pricing model, but also on the application behavior, which might not be fully known prior to production. A flat rate for data transfer, for example, might be a bargain for an I/O-intensive application, but a bad deal for a CPU-intensive one. Finding the cheapest price for cloud is certainly not as simple as shopping around for a deal on your electricity bill, where the actions of resellers and wholesale traders together with market regulation have made price comparison much less complicated.

As the market and technology evolve, electronic auctions will increasingly be used for real-time transactions for cloud capacity. Techniques have been proposed and prototypes built for matching, ranking, and truthfulness (that is, ensuring that users bid in accordance with true value of the service).[1]

However, one inherent challenge for such auctions to be realized is the substantial variation between cloud purchasing requirements as well as among CSPs' mechanisms for describing, metering, and charging for infrastructure services. The IEEE Intercloud standards efforts, such as the P2302 Standard for Intercloud Interoperability and Federation (https://standards.ieee.org/develop/project/2302.html), are intended to create uniform resource descriptions, ontologies (classification schemes), and semantics (resource meanings) to help facilitate such transactions. Even then there are nontechnical factors that will be challenging to incorporate, such as a buyer's trust in the service provider, or flexibility in accepting late payments. As a result, the primary customers of e-auctions might well be intermediaries, rather than end customers.

Moreover, for a customer to get the best overall price, as in real estate transactions where valuation is equally challenging, the reseller should be engaged as an agent for the buyer—representing and aligned with the cloud buyer's best interests—rather than an agent for the seller—acting on behalf of the CSP. Such a reseller needs to therefore be able to resell the services of multiple CSPs and to translate disparate pricing from these various CSPs so that they can be directly compared and contrasted and tradeoffs evaluated in light of the customer's specific needs. An example to watch is that of the University of Kentucky, which recently ran a competitive RFP for a financial cloud broker that would provide unified billing and pricing on an open-book basis from the winners of the (currently open) technical RFP for cloud IaaS services.

Rewarding Commitment

Most CSPs have initially offered pay-as-you-go, on-demand pricing to attract customers and in line with industry practice. However, to disincentivize customer defection due to lower pricing from a competitor, and to provide a better economic comparison with in-house private cloud deployments, CSPs have expanded their menu to offer discounts for prepayments. Prepayments should rationally be significantly lower than on-demand pricing for several reasons: to reflect expected future price cuts due, for example, to technology advances; to reflect the CSP's benefits in avoiding debt servicing; to reflect the time value of money; to eliminate the risk of default or nonpayment; and to reflect commitment to usage that de-risks capital investments in infrastructure.

It's this last discount where a reseller is truly able to add value and therefore take a margin, bridging the gap between customers, who are often reluctant to make commitments involving sizeable upfront payments, and CSPs, who benefit from these commitments. Of course, this bridging isn't without risks to the reseller. It's taking on credit risk because if the customer uses the CSP's services but fails to pay, the reseller won't receive a refund from the CSP. Another risk is that the customer might decide to no longer use the relevant cloud services, or use fewer of them. In such a case, if the prepayments aren't transferable, and the related cloud products are obscure or in an unusual geography, the reseller takes the loss. To minimize this risk, the reseller can encourage the customer to acquire a subscription for the specific services, committing the customer to pay for them in the future, whether or

This article originally appeared in
IEEE Cloud Computing, *vol. 2, no. 3, 2015.*

not they're actually utilized. Designing such a subscription to suit the customer is where a reseller can create substantial value and thus justify a margin. Such a margin is still subject to risk due to the not-unrealistic possibility of default, a possibility magnified by the wide range of customers served by most of today's providers.

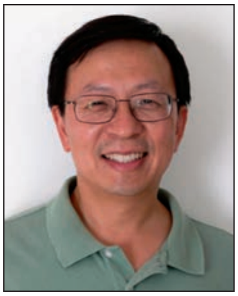
This isn't merely a theoretical analysis. For example, Amazon Web Services (AWS) recently launched one-year "no upfront" and "all upfront" heavy utilization reserved instances. By comparing the cash flows on these two offers, it's possible to ascertain the implied interest rate being charged by AWS for taking one year of financing risk. Such an analysis (available at www.strategic-blue.com/37pc) shows that the rate varies by service and by region, and is higher for less used services due to the reasons discussed previously. Moreover, even for a fairly common AWS instance like c3.large running Linux in US-East-1, the interest rate is a huge 36.7 percent per annum. This provides clear evidence of price reflecting the cost of risk, and some risks being deemed unacceptable—for example, there are no "no upfront reserved instances" with a three-year term.

IT RESELLERS AND DISTRIBUTORS ARE BEING FORCED TO TRANSFORM AS PART OF BROAD INDUSTRY RESTRUCTURING, OR RISK DISINTERMEDIATION BY UTILITY CLOUD SERVICES. They're also being forced to take on new financial risks, requiring new skills to manage and/or rely on specialist information and analyst services. The role of financial cloud brokers for multiprovider tenders for cloud services will continue to grow as customers realize that the new complexity of procurement, billing, and risk mitigation force the entire supply chain to adapt or crumble.

Reference

1. M. Anisetti et al., "E-Auctions for Multi-Cloud Service Provisioning," *Proc. IEEE Int'l Conf. Services Computing* (SCC 14), 2014, pp. 35–42.

JAMES MITCHELL *is CEO and cofounder of Strategic Blue (www.strategic-blue.com), a financial cloud broker offering billing, financing, and cloud price risk-management services through its Cloud Options and Pricing Insights services. He's a former commodities trader, and his research interests include the economics of utility services and the commoditization of IT. Mitchell has a doctorate in DNA nanotechnology from Oxford University. Contact him through his LinkedIn profile at https://uk.linkedin.com/in/jamesmitchellstrategicblue.*



The Growing Pains of Cloud Storage

Yih-Farn Robin Chen • *AT&T Labs Research*

Cloud storage is growing at a phenomenal rate, fueled by multiple forces, including mobile devices, social networks, and big data. Content is created anytime and anywhere on billions of smartphones and tablets; high-resolution photos and videos are frequently uploaded to the cloud automatically as soon as they're captured. A Gartner report predicts that consumer digital storage will grow to 4.1 zettabytes in 2016, with 36 percent of this storage in the cloud.[1] Social interactions and transactions on the Internet are frequently captured and analyzed for targeted advertising. In addition to social networks and e-commerce, big data analytics are growing in many other sectors, including government, healthcare, media, and education. An IDC forecast suggests that big data storage is growing at a compound annual growth rate of 53 percent from 2011 to 2016.[2]

The growth in cloud storage has made it an expensive cost component for many cloud services and today's cloud infrastructure. Whereas raw storage is cheap, the performance, availability, and data durability requirements of cloud storage frequently dictate sophisticated, multitier, geo-distributed solutions. Amazon Simple Storage Service (S3) offers 11 nines of data durability (99.999999999 percent), but some other services demand even more stringent requirements due to the sheer number of objects being stored in the cloud (1.3 billion Facebook users, uploading 350 million photos each day) and to the data's importance (who can afford to lose a video of their baby's first steps?). Data is frequently replicated or mirrored in multiple datacenters to avoid catastrophic loss, but copying it across datacenters is expensive. The networking cost is frequently proportional to the distance and bandwidth requirements between datacenter sites.

Traditional storage systems use dedicated hardware and networking to guarantee preservation of the quality-of-service (QoS) requirements, such as throughput, latency, and IOPS (total number of input/output operations per second). Unfortunately, these dedicated resources are frequently underutilized. Cloud computing promises efficient resource utilization by allowing multiple tenants to share the underlying networking, computing, and storage infrastructure. However, providing end-to-end storage QoS guarantees to individual tenants is difficult without mechanisms for avoiding interference. Typically, in a cloud environment such as Openstack, multiple tenants share the backend block storage (Linux's logical volume manager or a Ceph RADOS block device [RBD], for example) through a storage virtualization layer such as Cinder, which attaches virtual machines (VMs) to individual storage volumes. Providing customized storage QoS to meet different tenant needs is challenging. One exception is all-SSD storage arrays; some vendors (such as Solid Fire) let different tenants allocate storage volumes with different QoS types and dynamically change them, but all-SSD solutions (on the order of US$1,000 per terabyte) are expensive compared to HDD-based solutions. Moreover, an IOPS guarantee in the backend isn't sufficient because there might be contention for network bandwidth or CPU capacity from other tenants.

Finally, to operate any Web-scale solutions, infrastructure service providers are moving to scale-out solutions based on commodity hardware, instead of expensive storage appliances, which are frequently more expensive and difficult to adapt to changing workload or specific QoS requirements. Any cloud solution architect must understand the tradeoffs among the performance, reliability, and costs of cloud storage to provide an effective overall solution.

Emerging trends are sweeping through the storage industry to address these issues. Here,

 Published by the IEEE Computer Society 2376-113X/15/$31.00

I discuss two software-based solutions: erasure-coded storage and software-defined storage (SDS).

Erasure-Coded Storage

Erasure coding has been widely studied for distributed storage systems. Various vendors, companies, and open source software systems have adopted it recently, including EMC, Cleversafe, and Amplidata; Facebook, Microsoft, and Google; and Ceph, Quantcast File System (QFS), and a module of the Hadoop Distributed File System (HDFS-RAID), respectively. The primary reason for this adoption is that erasure-coded storage uses less space than fully replicated storage, while providing similar or higher data durability.

To understand why erasure coding is becoming crucial in storage systems, I must explain some basics. Erasure coding is typically controlled by two key parameters: k and n. A file or file segment is typically broken into k chunks, erasure coded, and expanded into n chunks ($n > k$) that are distributed over n storage servers or hard disks. Any k chunks are sufficient to reconstruct the original file, which can tolerate up to a loss of $m = n - k$ chunks without any data loss. One way to think about erasure coding is to consider a system of over-specified linear equations. You're essentially given n linear equations to solve for k variables. Picking any k out of these n equations would be sufficient to determine the values of those k variables. We frequently refer to the first k chunks as *primary chunks*, and the m chunks as *parity chunks*. Because we can vary k and m arbitrarily, a general erasure-coded storage solution in the form of (k, n) or $k + m$ has much higher flexibility in terms of the tradeoffs between storage space and reliability compared to the popular RAID 6 system, which uses only two parity blocks and is equivalent to a $k + 2$ erasure-coded scheme.

A scalable distributed storage system, such as HDFS or Swift, stored on multiple racks or sites typically uses triple redundancy (three copies of each data block) to improve both availability and durability. As the cloud storage volume continues to grow exponentially, the triple redundancy scheme becomes expensive. As an example, the QFS system uses 6 + 3 ($k = 6$ and $m = 3$) erasure coding and is designed to replace HDFS for MapReduce processing. HDFS uses triple replication and incurs 200 percent storage overhead, but it can only tolerate up to ANY two missing blocks of the same data. A 6 + 3 erasure code, on the other hand, can tolerate up to ANY three missing coded blocks with only 50 percent storage overhead. Such significant cost savings, while maintaining the same or higher reliability, is why many storage systems are now incorporating erasure codes.

One concern with erasure-coded storage is the extra overhead caused by the encoding/decoding time, which depends heavily on the erasure-coding scheme's strength. For a fixed k, higher m or n incurs more computation overhead while providing higher reliability. As computing servers gain in performance, the computation overhead of commonly used erasure codes becomes more manageable, and the bottleneck is frequently shifted to the disk or network throughput.

Another concern is the repair cost. Given that erasure coding of 6 + 3 requires six chunks to repair one chunk, the networking cost of repairing a chunk is six times that of a simple replication scheme. Some Facebook experiments use a 10 + 4 erasure-coding scheme, which incurs even higher repair costs (but lower storage overhead at 40 percent). Several repair schemes (such as Xorbas[3] and Hitchhiker[4]) have been proposed to reduce the repair bandwidth, with or without additional storage overhead.

As data durability becomes increasingly important for cloud storage, erasure coding can also play an important role in cloud storage geo-distribution. It allows chunks of an erasure-coded file to be placed in multiple datacenters or racks to increase data durability. For example, a 9 + 15 or (9, 24) erasure-coded storage system could put six chunks each in New Jersey, Illinois, Texas, and California (east, north, south, and west areas of the US). Because any file can be reconstructed from nine chunks, it can be reconstructed by retrieving those chunks from any two datacenters. Thus, it will tolerate up to two datacenter failures, even during a major natural disaster such as 2013's Hurricane Sandy, which caused a loss of 68 billion dollars and affected 24 states. On the other hand, because each file retrieval requires accessing chunks from two datacenters, it might incur longer latency and significant communication costs, which is fine for archival storage, but not ideal for frequently accessed storage. Alternatively, if we know certain files' access patterns, and it turns out that most accesses come from New Jersey, we can place nine chunks in New Jersey and five chunks each in Illinois, Texas, and California. This would allow users to complete most accesses with low network latency and slightly lower reliability, given that a datacenter loss has the potential to lose nine instead of six chunks. The chunk-placement issue in erasure coding affects latency, cost, and reliability in geo-distributed storage systems and is currently an active research field.

To realize the vision of a virtualized datacenter, we need software-defined storage that virtualizes storage resources and separates storage management software from the underlying hardware.

Software-Defined Storage

Cloud computing started with the virtualization of computing resources, followed by recent advances and rapid innovations in software-defined networks (SDNs), which aim to virtualize networking resources and separate the control plane from the data plane. To truly realize and complete the vision of a virtualized datacenter, however, we need software-defined storage that virtualizes storage resources as well and separates storage management software from the underlying hardware.

Unfortunately, unlike SDN, there isn't a clear definition of what software-defined storage really is, although many storage vendors claim that they have SDS solutions. Most SDS definitions include a list of desirable attributes.[5,6] Here, I summarize those that pertain to multitenant cloud storage solutions, what I call the S.C.A.M.P. principles of SDS.

Scale-Out

SDS should enable a scale-out (horizontal scaling of low-cost, commodity hardware) instead of a scale-up (vertical scaling using more powerful hardware) storage solution as the workload grows or changes dynamically over time. A scale-out solution is best implemented in a cloud environment with large computing, networking, and storage resource pools. A cloud storage solution is never just about storage — all the necessary computing and networking resources must also scale accordingly to support common storage operations: deduplication, compression, encryption/decryption, erasure coding/replication, and so on.

Customizable

SDS should allow storage system customization to meet specific storage QoS requirements. This lets customers purchase storage solutions based on their specific performance and reliability constraints and avoid unnecessary over-engineering, which frequently happens when a cloud storage service provider tries to meet the needs of multiple customers with diverse requirements. In a multitenant cloud with a shared backend storage, guaranteeing the desired storage QoS is particularly difficult. The latest version of Openstack Cinder, which provides a block storage service, now allows multiple backends with different QoS types (such as different IOPS or throughput numbers) to partially address this issue.

Automation

Once storage QoS requirements are clearly defined, SDS should automate the complete provisioning and deployment process without human intervention. The current practice is that a storage architect or system administrator is intimately involved in designing and installing the storage system. This process is typically error-prone and not amenable to adapting to changing workloads or requirements in real time.

Masking

SDS could mask the underlying storage system (physical or virtualized) and distributed system complexity (single or multiple-site) as long as such systems can present a common storage API (block, file system, object, and so on) and meet QoS requirements. This gives infrastructure service providers greater flexibility in restructuring their resource pools or architecting storage systems. For example, Ceph can present a block device API even though the underlying implementation is done in its RADOS object storage.

Policy Management

SDS software must monitor and manage the storage system according to the specified policy and continue to meet storage QoS requirements despite potential interference from other tenants' workloads. It must also handle failures and autoscale the system when necessary to adapt to changing workloads. As stated previously, however, guaranteeing end-to-end storage QoS in a multitenant cloud is a hard problem that requires protecting resources on the entire path from a VM to the storage volume. Microsoft's IOFlow[7] aims to provide an SDN-like controller to control storage bandwidth allocation at multiple points of such a path.

SDS Definition

By combining the S.C.A.M.P. principles, we can now define SDS: an SDS solution should automatically map customizable storage service requirements to a scalable and policy-managed cloud storage service, with abstractions that mask the underlying storage hardware and distributed system complexities.

Incidentally, erasure coding is a crucial technology that can help meet the SDS customization requirement. For a fixed k, varying n (or m, the number of parity chunks) increases the reliability and replication factor (and hence the storage cost). At the same time, it increases the overall encoding/decoding time, hence the required computation capacity, and perhaps reduced performance. This lets an automated storage architect look at the storage QoS requirements and pick particular erasure-code parameters (k and m) to meet the minimal reliability and performance requirements with the least amount of storage overhead.

The rapid growth of cloud storage has created challenges for storage architects to meet different customers' diverse performance and reliability requirements while controlling costs in a multitenant cloud environment. Erasure-coded storage and SDS could address these challenges and open up new opportunities for innovation. Moreover, erasure coding could play a crucial role in offering design tradeoffs in certain SDS solutions. These two technologies, working together, have a huge potential to address the growing pains of cloud storage and help ease the transition from traditional IT storage solutions — given that cloud storage will likely support a large portion of all IT storage needs in the future.

References

1. "Gartner Says that Consumers Will Store More than a Third of Their Digital Content in the Cloud by 2016," Gartner, press release, 25 June 2012; www.gartner.com/newsroom/id/2060215.
2. "Big Data Drives Big Demand for Storage, IDC Says," *Business Wire*, 16 Apr. 2013; www.businesswire.com/news/home/20130416005045/en/Big-Data-Drives-Big-Demand-Storage-IDC.
3. M. Sathiamoorthy et al., "XORing Elephants: Novel Erasure Codes for Big Data," *Proc. VLDB Endowment*, 2013, pp. 325–336.
4. K. Rashmi et al., "A Hitchhiker's Guide to Fast and Efficient Data Reconstruction in Erasure-Coded Data Centers," *Proc. 2014 ACM Conf. SIGCOMM*, 2014, pp. 331–342.
5. B. Earl et al., "Software-Defined Storage," *5th Usenix Workshop on Hot Topics in Storage and File Systems*, panel, 2013; www.usenix.org/conference/hotstorage13/workshop-program/presentation/earl.
6. M. Carlson et al., "Software-Defined Storage," Storage Networking Industry Assoc. working draft, Apr. 2014; http://snia.org/sites/default/files/SNIA%20Software%20Defined%20Storage%20White%20Paper-%20v1.0k-DRAFT.pdf.
7. E. Thereska et al., "IOflow: A Software-Defined Storage Architecture," *Proc. 24th ACM Symp. Operating Systems Principles*, 2013, ACM, pp. 182–196.

Yih-Farn Robin Chen is a Lead Inventive Scientist at AT&T Labs Research. His research interests include cloud computing, storage systems, mobile computing, and distributed systems. Chen received a PhD in computer science from the University of California, Berkeley. He's a vice chair of the International World Wide Web Conferences Steering Committee (IW3C2) and a member of the editorial board of *IEEE Internet Computing*. Contact him at chen@research.att.com.

This article originally appeared in IEEE Internet Computing, *vol. 19, no. 1, 2015.*

Socioeconomics of Cloud Standards

LET'S START THIS COLUMN WITH A DISCLAIMER: MY COLLEAGUE JOE WEINMAN ALREADY DOES A TERRIFIC JOB OF COVERING TOPICS RELATED TO THE ECONOMICS OF CLOUD COMPUTING IN THE DEPARTMENT DEVOTED TO THAT AREA IN THIS MAGAZINE. There are, however, several ways in which the economics of standards can differ from those of a field in general, and this is true also for the field of cloud computing. There are also specific sociological factors that influence the adoption of technologies in areas like cloud computing that depend on aggregating input from multiple participants.

In this instance of the column, I discuss factors governing the growth of consensus on technologies in ways that can lead to standards, and try to explain features that can both limit their initial emergence and, in successful cases, accelerate their rapid subsequent uptake. Most of these features also apply to development and adoption of software, especially in open source and other shared-development settings. There are, however, some unique factors at play that influence development of standards slightly differently from most software. I'll point out conditions that tend to lead to monopolies versus dynamics of shared development and diversity that push communities in the opposite direction toward standards.

ALAN SILL

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

Conditions for Emergence of Standards

The special type of development that leads to the creation and adoption of standards differs from other forms of technological development in specific ways. First of all, standards are only needed if a plurality of options exists within a given area that makes selecting a specific solution from among these both necessary and possible. It makes no sense to describe a situation characterized by a complete, absolute monoculture using the terminology of standardization. If the number of available choices for a given technology or approach is exactly one, the right term to use is "monopoly."

The number of choices shouldn't be confused with the technological diversity among those choices. Many vendors of the same brand of a given product, say a pack of gum, still give you only one choice of the type of gum to chew. The same situation applies to hardware and software. A monopoly consisting of a single supplier with multiple points of sale is still a monopoly. The diversity among the technological choices also influences whether standards are needed.

The complexity of the range of choices available to users also plays a role. Standards aren't needed, and in general don't emerge, if the set of choices is so broad, uniquely selectable, and intrinsically self-contained that each decision will work as well as any other, with no subsequent resulting consequences. A standard makes sense, as I've discussed in previous columns, if an overall simplification in implementing the chosen solution results from making a given technology choice. The process of selecting among the available options should thus produce a more convenient, more interoperable, or generally more manageable landscape in order for standards to emerge.

The most basic factors that govern the emergence of standards within a given community are

Published by the IEEE Computer Society

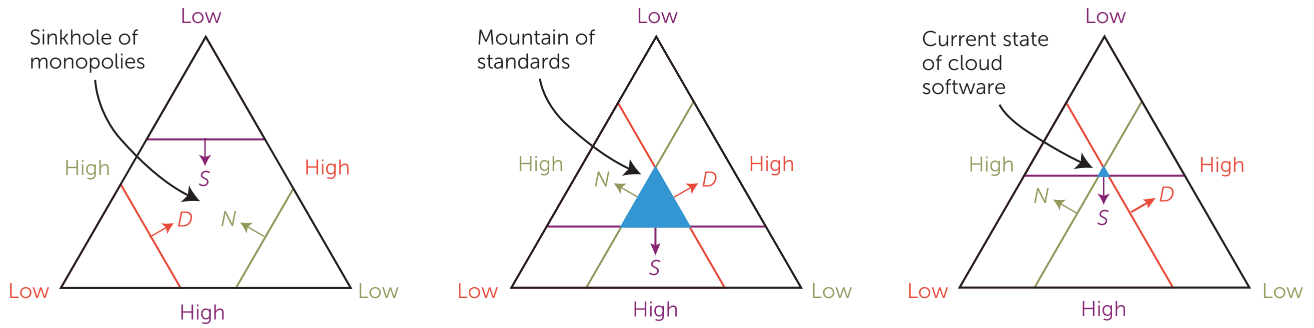


FIGURE 1. The number (*N*), diversity (*D*), and simplicity (*S*) characterizing conditions that affect emergence of common consensus-based standards as opposed to the isolation of solution options that lead to monopolies within a given technology-based community can be related to each other through the use of this triangular NDS diagram.

thus the number of available choices, the diversity among technologies offered by these choices, and the simplicity of the solution space among these choices.

A Simple Rubric: The NDS Diagram

These considerations are quantifiable and suitable for numerical analysis. They can, for example, be plotted against each other on a diagram, which I present as triangles in Figure 1. This isn't the only way to perform this type of analysis, and other axes could be used, but for now let's keep to this simple representation.

In this diagram, we can represent the number (*N*), diversity (*D*), and simplicity (*S*) of solutions in a given technology space as the distance from the point to the broad side of the triangle in each case, as these variables grow from small to large. I'll leave off the mathematical description that can be built from these variables, and discuss them in purely relative qualitative terms.

A solution with only one supplier would mean that the green line depicting *N* would be all the way at the lower-right corner of the triangle. A situation with low diversity among solution providers would limit the red line for *D* to the lower-left corner. A situation with low simplicity (that is, a very complex solution space that requires great expertise with a long learning curve) would keep the blue line for *S* up at the top of the triangle.

This situation, in which any or all of the number, diversity, or simplicity variables is very small, tends to lead to domination of a field by a small, nondiverse or complex set of providers—in other words, toward monopolies. I label this gap the "sinkhole of monopolies" in the diagram.

If, on the other hand, the number, diversity, and ease of adoption of solutions among the range of providers all become sufficiently large that these lines move away from their low-value individual vertices toward the broad sides of the triangle, they can overlap to create a region that I refer to as the "mountain of standards" in the diagram. It's under these conditions, in which multiple suppliers provide a range of choices that can be simplified by making a selection among them, that standards are needed and tend to emerge.

Standards Breakout and Uptake

This might seem like a complex way to look at a simple problem, but it helps us to understand several characteristics of standards development that are otherwise hard to comprehend and easier to see with the aid of this diagram.

Consider the current situation for cloud computing, which I designate in the third triangle in Figure 1. Only a small (but growing) number of standards have garnered significant attention from the developer community so far. The number of cloud providers, however, continues to grow. In terms of the variables of this diagram, *N* is becoming larger, even if you count only very large providers.

Similarly, the diversity *D* of the provider space continues to increase. Although repeated analyses of infrastructure-as-a-service (IaaS) cloud providers show market dominance by Amazon Web Services,[1] complete cloud solutions that don't use AWS interface methods are being offered successfully by an increasingly diverse set of companies, such as Microsoft, Google Cloud, and the open source OpenStack-based set of providers, with significant market uptake. This increasing diversity of the viable set of providers also tends to push in the direction of standards emergence.

Finally, the simplicity of deploying cloud solutions is improving rapidly.

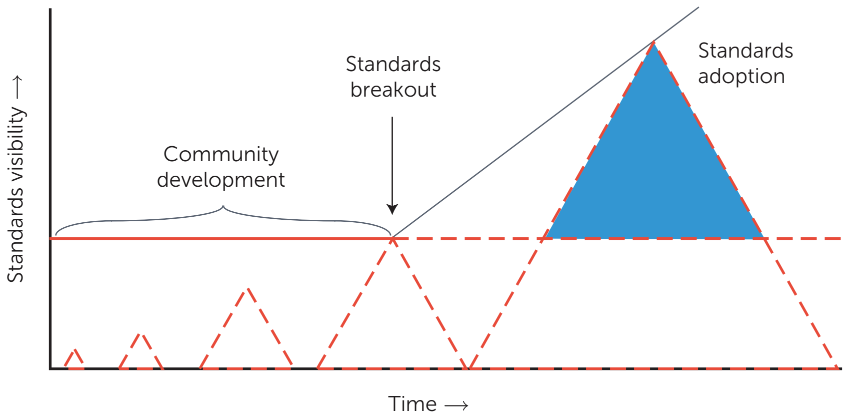


FIGURE 2. A period of community development typically precedes the emergence of an overlap region sufficient to produce the conditions that lead to breakout of a standard into widespread visibility. Once this overlap occurs, as Figure 1 illustrates, adoption can occur quickly.

Because cloud-based solutions cover a broad range of technological situations, and innovations have arrived at a dizzying pace, the solution space has been complex. Making a particular choice among cloud-based solutions up to now hasn't necessarily been a way to guarantee simplicity, or even survivability, of a given decision. The *S* variable in the NDS diagram (Figure 1) has thus been kept to the small apex at the top of the triangle. It has taken a long time compared to this pace of innovation to move to a situation, which I believe is now the case, in which choosing from among available cloud methods can result in an overall simplification of a given business or community technology problem.

When this situation results, in which the *N*, *D*, and *S* variables in a given community space all move to larger values, we reach conditions in which standards use can break out rapidly, and adoption can grow at a pace that would surprise anyone who isn't following all of these factors. Figure 2 illustrates this situation. In evaluating whether or not this will occur in any given situation, the socioeconomic factors that limit the rate of growth of number, diversity, and simplicity of choices must all be considered.

Of course, other conditions can conspire to keep an area of technology pinned toward one or more of the vertices of this triangle and away from the mountain of standards that emerges when they overlap in the middle. I am willing to argue, however, that the situation for cloud computing at present is more like the one shown in the third triangle in Figure 1, and that the pressures are very strong at present for at least one of these variables—that of simplicity—to grow significantly in the near future.

Additional Factors

To complete a discussion on cloud standards socioeconomics, we also need to consider the cost benefits of aggregation around a given cloud technology, such as a standard, and compare these to the market-based effects of the reduction in costs that can occur if a given dominant player not using that standard or technology were simply to acquire more customers.

If a breakout like that shown in Figure 2 doesn't occur naturally, in some cases interventions could take place, either through government regulation or through the emergence of another, externally driven or sociologically motivated set of choices. It's for this purpose that market regulators emerge, often through government intervention or regional market protections, to drive the number and/or diversity of providers to larger values.

The opportunity for disruptive technology innovations to emerge to offer simpler solutions is also a factor, although a simpler solution in a given market subset can also increase the overall complexity (that is, lower the overall total simplicity) of the market arena. Thus, there might not be only one diagram of the sort shown in Figure 1 for a given community or problem. Instead, there could be other variables beyond number, diversity, and simplicity that need to be factored in.

The presence of other factors is a complication, but the basic style of analysis presented here can still be adapted to deal with these other features. The *N*, *D*, and/or *S* variables could be replaced, or additional variables added to turn the triangle shown in Figure 1 into a more complex polyhedron. A complete analysis might require breaking the technology space into subsets to see whether the overlap ("mountain") or technology gap ("sinkhole") conditions hold in each particular area. Nonetheless, the basic idea presented here should still hold, which is that a period during which technologies mature and diversify generally needs to take place before a single idea can create conditions under which standards emerge, or are even needed.

Cloud Standards Conclusions

My basic point is the following: cloud computing has been around long enough now that it has attracted a large enough

This article originally appeared in
IEEE Cloud Computing, *vol. 2, no. 3, 2015.*

number of major players with significant enough diversity among them that standards can be expected to emerge. It's specifically not the case that standards creation has been inhibited by the presence or domination of large companies, nor is it the case that large open source projects can by themselves be expected to lead to standards-based solutions. Instead, simultaneous or at least overlapping conditions need to occur in the number, variety, and ease of use of cloud technologies before standards can emerge and come into widespread use.

The main factor that has been limiting the emergence of cloud standards to date has been the rapidly changing nature of the technology, and thus the low degree of technological simplification that can result from making standards-based choices among these technologies that themselves lead to interoperability and consumer choices among providers. This relatively slow rate of progress toward overall simplification of the solution space, which is needed for standards to emerge, has itself been driven by the rapid pace of creation of other related technologies that has kept even the "800-pound gorillas" off-balance and searching for stable technologies on which to base their offerings.

If my interpretation is accurate, then as time goes on, we should expect cloud standards such as those I discuss in an earlier column[2] to grow, and the potential exists for one or more of them to experience rapid growth in the near future. The main socioeconomic factor that's hard to anticipate is whether the cost-lowering or community-growing effects of aggregation around adoption of cloud standards can outweigh the advantages of scaling the use of one or more of the individual dominant market providers.

Based on activity on open source code repositories such as GitHub, and community adoption patterns that are leading to new companies and software projects, I think the situation is one in which the overall simplification of cloud technologies will win. If so, we'll see new patterns emerge fairly soon that use cross-cutting specification-based technologies—that is, standards. Future "Standards Now" columns and articles in the standards area will attempt to provide a summary of such activities.

THE ANALYSIS PRESENTED HERE IS UNDOUBTEDLY AN OVERSIMPLIFICATION, AND IS OFFERED MERELY BY A PARTICLE PHYSICIST WITH A BACKGROUND IN COMPUTING, NOT BY A TRUE ECONOMIST OR SOCIOLOGIST. What do you think are the important factors governing technology development and standards adoption in cloud computing? Where do you think this analysis can be improved?

Please respond with your opinions on this topic or on previous columns, and please also include any news you think the community should know about the general areas of cloud standards, compliance, or related topics. As mentioned earlier, we're always open to article submissions that cover these topics. I'll be happy to review ideas for such submissions and can be reached for this purpose at alan.sill@standards-now.org.

References

1. L. Leong et al., *Magic Quadrant for Cloud Infrastructure as a Service, Worldwide*, Gartner, 18 May 2015; www.gartner.com/doc/3056019/magic-quadrant-cloud-infrastructure-service.
2. "Cloud Standards News and Updates," *IEEE Cloud Computing*, vol. 2, no. 1, 2015, pp. 72–75.

ALAN SILL *directs the US National Science Foundation Center for Cloud and Autonomic Computing site at Texas Tech University, where he's also a senior scientist at the High Performance Computing Center and adjunct professor of physics. Sill has a PhD in particle physics from American University and is an active member of IEEE, the Distributed Management Task Force, TeleManagement Forum, the Open Grid Forum and other cloud standards working groups, and serves on several national and international standards roadmap committees. For further details, visit http://cac.ttu.edu or contact him at alan.sill@standards-now.org.*

Software Modernization Revisited: Challenges and Prospects

Hugo Bruneliere, Inria—Mines Nantes—LINA

Jordi Cabot, ICREA—Universitat Oberta de Catalunya

Javier Luis Canovas Izquierdo, Universitat Oberta de Catalunya

Leire Orue-Echevarria Arrieta, Tecnalia

Oliver Strauss, Fraunhofer IAO

Manuel Wimmer, Vienna University of Technology

The authors discuss important factors to consider when migrating software to the cloud and offer recommendations to maximize the chance of success.

Organizations undertake more and more software modernization projects every day, mostly owing to rapid changes in the technological landscape pushing them to evolve their systems before those systems become obsolete. Such projects are sometimes taken too lightly and start more because of passing fads than because they're motivated by real technological limitations or system problems. In our experience, many managers have only a partial view of these projects' complexity and consequences.

A modernization project usually has three phases. First, reverse engineering provides an understanding of the system's purpose and current state. Many supporting approaches rely on model-based techniques to discover software models representing the system at a higher abstraction level. Examples of such models are UML (Unified Modeling Language) class diagrams, state machines, and workflows. Second, forward engineering analyzes those models and transforms them (if

 2376-113X/15/$31.00 © 2015 IEEE

EDITOR **MIKE HINCHEY**
Lero—the Irish Software Research Centre;
mike.hinchey@lero.ie

necessary) to become the modernized system's specification. Finally, developers or automated code generation techniques (or a combination of both) use these models to produce the corresponding code for the targeted platform.

Tools can help in some of these tasks—for example, tools that discover UML models from Java code, or code generators for various platforms. But currently no global methodology exists that can guide software engineers through the whole process. Moreover, no real support exists for assessing project risks or final software quality.

On the basis of our experience in conducting and building tools for software modernization projects, we discuss here important factors to consider in such projects and offer recommendations to maximize the chance of success.

THE ARTIST PROJECT

We follow the approach taken by the ARTIST project (Advanced Software-Based Service Provisioning and Migration of Legacy Software; www.artist-project.eu). ARTIST is an ambitious European collaboration involving industry (the Athens Technology Center [ATC], Atos, Engineering Ingegneria Informatica, Sparx Systems, and Spikes) and academic institutions and research centers (Fraunhofer IAO, the Institute of Communication and Computer Systems [ICCS], Inria, Tecnalia, and the Vienna University of Technology). Part of the EU's Seventh Framework Programme for Research, the project started in October 2012 and is scheduled to last three years. It proposes a complete open-technology and vendor-neutral approach to software migration to cloud environments, including a generic methodology and base tooling support.

THE REAL FACE OF SOFTWARE MIGRATION

To get a representative vision of the factors impacting software modernization projects, we analyzed four case studies (from ARTIST industrial partners) of the migration of real software systems to the cloud:

- DEWS CCUI (Distant Early Warning System for Tsunamis Command and Control User Interface, by Atos), a Java system for tsunami early detection;
- LoB (Line of Business, by Spikes), a .NET solution for business process management;
- eGov (by Engineering Ingegneria Informatica), a J2EE (Java 2 Enterprise Edition) framework to support Italian citizens' interaction with their government; and
- News Assets (by the Athens Technology Center), a .NET news publication management system.

After initial analysis, we came up with six dimensions that helped us systematically evaluate each project's complexity:

- *Technical space (TS)*. A TS is a family of knowledge, tools, and technologies used to implement the system. For instance, some software artifacts belong to the grammarware TS (for example, source code files and the grammars they conform to), and others belong to the xmlware TS (for example, XML-based configuration files and their document type definition or schema). Each TS imposes a different reverse-engineering approach.
- *Origin*. Some software artifacts are generated; others are manually created (for example, the scaffolding of project files derived from specifications). It's important to filter out generated artifacts for which their abstract specification is available.
- *Purpose*. We identified main categories—for example, code or application, data, configuration, specification, and documentation. Having a fine-grained artifact categorization is fundamental for engineers to prioritize or filter artifacts during migration.
- *Architectural viewpoint*. We observed a typical four-layer classification: presentation, business logic, data, and communication. Again, this helps in problem decomposition and in identifying key company stakeholders who can assist during migration (depending on the layers to which the artifacts belong).
- *Environment*. External technological requirements and dependencies can strongly influence the process.
- *Size*. Size (for example, of memory) and the number of individual components are key elements to consider and study.

We circulated a survey among the case study companies and conducted interviews to evaluate each project regarding the six dimensions. Table 1 summarizes the results at the project level, even though our analysis was at the artifact level. Getting a better picture of the main project challenges (for example, each project had to treat and properly integrate a mix of TSs) benefitted both the companies and us. We recommend that companies perform a similar analysis before starting any migration project.

KEY SUCCESS FACTORS

From our experiences with the ARTIST project, we found that the following four key factors contribute to a more

TABLE 1. The four case studies according to the identified dimensions.

Dimension	DEWS CCUI, Atos	LoB, Spikes	eGov. Engineering Ingegneria Informatica	News Assets, Athens Technology Center
	Case study project and company*			
Technical space	Source code (Java, Python), XML	Source code (C#, PowerShell JavaScript, HTML, CSS [Cascading Style Sheets], ASP XAML [Active Server Pages Extensible Application Markup Language), XML, plain graphics	Source code (Java, OWL [Web Ontology Language], WSDL [Web Services Description Language]), XML, plain text, plain graphics	Source code (C#, JavaScript, HTML, CSS), XML
Origin	Mainly manual code, some code generation	Mainly manual code, some code generation based on domain-specific languages (DSLs)	Balanced (a partly generative approach for code)	Mainly manual code, some code generation
Purpose	Application, data	Application, configuration	Application	Application, data, configuration
Architectural viewpoint	Presentation, business logic, data	Presentation, business logic, data	Presentation, business logic, data	Presentation, business logic, data, communication
Environment	Eclipse Platform (Java), Linux	Microsoft Visual Studio, SQL Server (.NET), Windows	Eclipse Platform (Java), Protégé (ontologies)	Microsoft Visual Studio (.NET), Oracle relational database management system
Size	Medium	Medium for GPL parts, rather small for DSL parts	Large for ontology parts, rather small for the rest	Large for the application, medium for the rest

* DEWS CCUI: Distant Early Warning System for Tsunamis Command and Control User Interface; LoB: Line of Business.

unified, focused, goal-oriented, and guided migration.

One format to rule them all

Model-based migration, in which the system is represented by interrelated models (showing different system views), lets us more easily deal with the TSs' heterogeneity. Specialized components (discoverers) inject the content of artifacts (from different TSs) into the common model-based migration platform (into the modelware TS). Discoverers for several kinds of grammar-based and XML-based artifacts are publicly available. For the generated models to be interoperable, the modeling platform shares a common meta-metamodel (provided by the Eclipse Modeling Framework in our case).

We've benefited most from an appropriate mix of modeling languages such as

- UML,
- SysML (Systems Modeling Language),
- BPMN (Business Process Model and Notation),
- KDM (Knowledge Discovery Metamodel, part of the Architecture-Driven Modernization initiative; http://adm.omg.org), and
- some domain-specific languages (DSLs).

We also rely on corresponding model-based techniques, such as model transformation or text and code generation, to process models generically.

Different views for different stakeholders

Modeling frameworks separate the visualization of modeled information (the concrete syntax) from its content (the abstract syntax). This lets engineers process model content efficiently and provides several visualizations (possibly using alternative notations) of a model or model fragment. This distinction is especially useful for general-purpose modeling languages such as UML or ODP (Open Distributed Processing) that predefine different viewpoints for different stakeholders. It lets engineers compute several views of a system to emphasize or disregard certain aspects.

From our experience, viewpoints are beneficial in migration projects, and not only for standard system design. During reverse engineering, viewpoints help improve system understanding (for example, separating a system's structure from its behavior, or the architecture from a detailed implementation). In forward engineering, viewpoints are fundamental for making decisions on software modifications.

Nonfunctional properties as first-class citizens

Although migration projects traditionally focus on systems' functional aspects, they also deal more frequently with nonfunctional aspects. This is particularly true for software migration to the cloud. Often, the base platforms and programming languages don't change during migration. However, cloud environments might introduce features or capabilities that can help improve nonfunctional properties that are important for the system and its owner. In such projects, nonfunctional properties are the main driver for migration and must be taken into account in all phases. A concrete example is the design and realization of refactorings tailored to improve performance and scalability.

Migration as a process

Besides representing software artifacts, models also help explicate knowledge of the migration process. Engineers can exploit this to enact a well-defined migration process for their projects.

We've established a systematic road map for achieving software migration by providing a reference process (as an explicit process model) to be customized for specific migration projects. The available tooling supports this customization and allows following the specified process step-by-step. So, defined processes not only concentrate on actual code migration activities but also address premigration (for example, specifying migration goals and identifying a target environment to satisfy expressed requirements) and postmigration (for example, evaluating initial migration goals against the finally migrated system).

A CONCRETE APPROACH TO CLOUD MIGRATION

Using the principles and guidelines we've been describing, we developed a concrete methodology as part of ARTIST (see Figure 1). As summarized graphically, it has three phases:

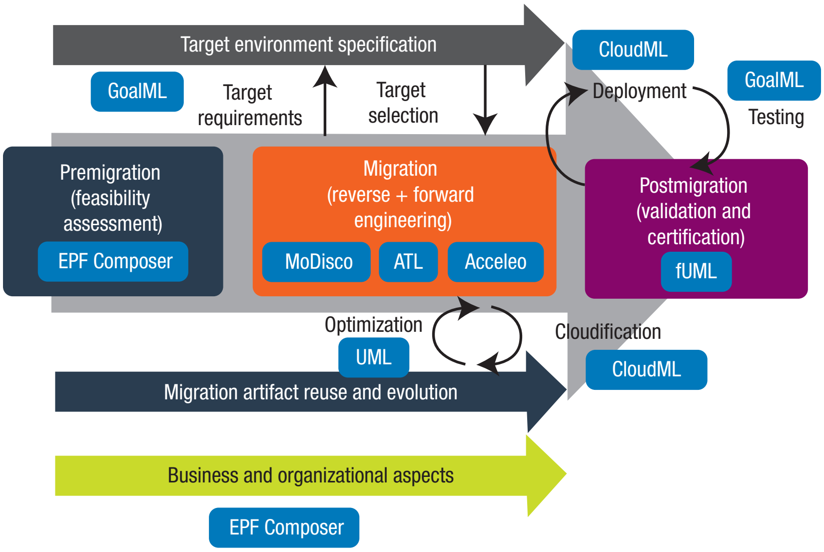


Figure 1. The ARTIST (Advanced Software-Based Service Provisioning and Migration of Legacy Software) process. This migration approach intends to be an answer to the challenges discussed in this article, and a concrete application of the success factors we've identified.

- Premigration evaluates the existing software, notably in terms of migration feasibility from both a technical and business perspective (for example, checking cloud compliance on the basis of standards such as the Topology and Orchestration Specification for Cloud Applications [TOSCA] or drafts such as the Cloud Computing Reference Architecture [CCRA]).
- Migration covers both reverse- and forward-engineering activities (including possible optimizations in the migrated software). It also includes selection of the target cloud provider.
- Postmigration addresses the migrated software's verification and validation. It also addresses the potential certification of the newly produced (cloud-based) pieces of software, notably in regard to cloud standards and current best practices.

We've implemented two versions of the tooling to support this methodology. One relies on the Eclipse framework and handles Java/J2EE cases. The other is based on Sparx Systems Enterprise Architect and handles C#/.NET cases. Throughout the process, and particularly during migration and optimization, we extensively use UML models (sometimes extended with some profiles) to represent the system's required aspects. This also facilitates exchanging models between tools and building several viewpoints. All used or produced models can be stored in or retrieved from the ARTIST repository at any time.

We employ other modeling languages and techniques in more domain-specific tasks. For example, we use GoalML (Goal Modeling Language) during premigration to precisely determine the migration's scope (for example, regarding nonfunctional properties) and to help determine the migration's feasibility. During postmigration, fUML (Foundational UML) behavioral models of the deployed system are automatically checked against the initial goal models to validate that the corresponding objectives are satisfied. The target specification and deployment process depend on

CloudML (expressed as a UML profile) to represent the required cloud infrastructures.

This is made easier by the reuse of open source modeling solutions (thanks to the shared modelware TS), mainly from the Eclipse ecosystem. In particular, we employ the Eclipse Process Framework (to enact and customize the ARTIST Methodology Process Tool [MPT] to follow the various ARTIST phases), the MoDisco model-driven reverse-engineering framework (for code-to-model discovery), ATL (for model-to-model transformation), and Acceleo (for model-to-code generation).

In ARTIST we elaborated on a complete migration methodology and base supporting tooling. But obviously, aspects of the different steps of the process still need improvement. So, we're creating an ARTIST Club, which will initially involve the project's partners and later be open to external participants. We want to make this collaborative effort a good opportunity to continue capitalizing on the ARTIST results, in terms of future research achievements (for academics) and further commercial exploitation (for industrial partners).

HUGO BRUNELIERE is a research engineer and project manager on the AtlanMod Team, a joint Inria, Ecole des Mines de Nantes, and LINA (Nantes Atlantic Computer Science Laboratory) research team. Contact him at hugo.bruneliere@inria.fr or hugo.bruneliere@mines-nantes.fr.

JORDI CABOT is an ICREA (Catalan Institution for Research and Advanced Studies) research professor at Universitat Oberta de Catalunya (UOC) and previously led the AtlanMod Team. Contact him at jordi.cabot@icrea.cat.

JAVIER LUIS CANOVAS IZQUIERDO is a postdoctoral fellow at UOC. He previously was on the AtlanMod Team. Contact him at jcanovasi@uoc.edu.

LEIRE ORUE-ECHEVARRIA ARRIETA is a project manager and the leader of the Cloud Technologies Team in Tecnalias's ICT-European Software Institute Division. Contact her at leire.orue-echevarria@tecnalia.com.

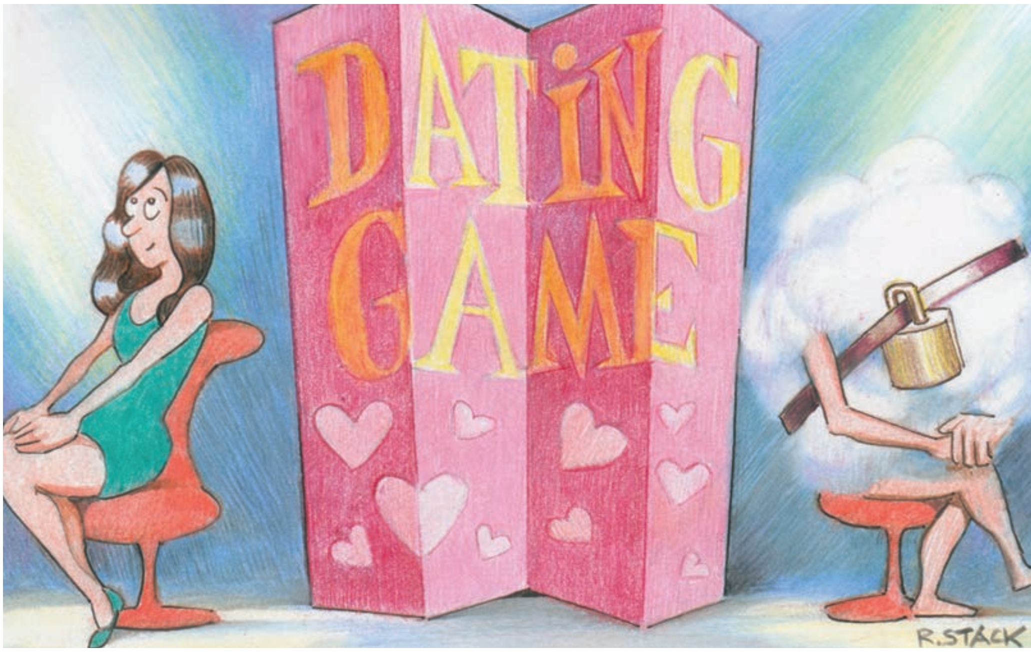
OLIVER STRAUSS is a research engineer on Fraunhofer IAO's Software Technology Competence Team. Contact him at oliver.strauss@iao.fraunhofer.de.

MANUEL WIMMER is a senior researcher in the Vienna University of Technology's Business Informatics Group. Contact him at wimmer@big.tuwien.ac.at.

This article originally appeared in Computer, *vol. 48, no. 8, 2015.*

Confidentiality in the Cloud

Ricardo Padilha and Fernando Pedone | University of Lugano, Switzerland



Cloud computing, formerly heralded as the solution to all large-scale computing problems, has recently lost some appeal. The constant stream of privacy breaches, either from government agencies or hackers of cloud-hosted data, has caused an understandable skepticism toward cloud computing as a platform capable of preserving privacy and confidentiality. We contend that the future of cloud computing requires fundamental changes in the way large-scale distributed systems are designed to account for data confidentiality.

Failure Model

Traditional fault-tolerant systems assume a failure model, where faulty components such as processes, servers, and networking equipment stop responding once a fault occurs. This assumption doesn't take into account agents whose goal is to corrupt data silently (at rest or in transit), or to observe and track users rather than disrupt a service.

The *Byzantine fault-tolerant* (BFT) model makes fewer assumptions about faulty behavior.[1] In the Byzantine model, faulty components can present arbitrary behavior and continue to interact with other components even after a failure, often producing faulty data and corrupting their state. In short, the only limitation on the behavior of faulty components is that they can't prevent correct components from following their proper and expected behavior. BFT systems can tolerate up to f Byzantine replicas by replicating data in $2f + 1$ replicas. In addition, BFT systems that implement state-machine replication require $3f + 1$ replicas to agree on the sequence of commands.

With a judicious choice for f, BFT systems prevent the silent corruption of data. However, this isn't sufficient to provide confidentiality. A Byzantine (or compromised) server can still monitor, collect, and track users' data. For example, even when all data is encrypted, a Byzantine server can leak contextual information such as access records through the delay between replies.

We can achieve confidentiality in BFT systems through either obfuscation or firewalling (see Figure 1). We use *obfuscation* as an umbrella term for methods that protect the plaintext by a reversible yet practically unbreakable permutation, and not as a synonym for steganography, which is a much weaker method of obfuscation. Each of these approaches presents a different set of tradeoffs, which are shown in Table 1. In the "Round-trip latency" column, δ indicates the delay between clients and servers, δ_C indicates the communication delay of the consensus algorithm, and δ_F indicates the communication delay of the privacy firewall.

Table 1. Confidentiality in Byzantine fault-tolerant systems.

Type	Supported operations	Round-trip latency	Storage	Infrastructure	CPU usage	Management	Example
Firewall	Arbitrary	$\delta_C + \delta_F + 2\delta$	$O(1)$	Large	High	Simple	Jian Yin and his colleagues, 2003
Cryptography	Storage	$\delta_C + 2\delta$	$O(1)$	Small	Low	Complex	Ernesto Damiani and his colleagues, 2003
Secret sharing	Storage	$\delta_C + 2\delta$	$O\left(\frac{n}{f+1}\right), O(n)$	Small	Low	Simple	Ricardo Padilha and Fernando Pedone, 2011

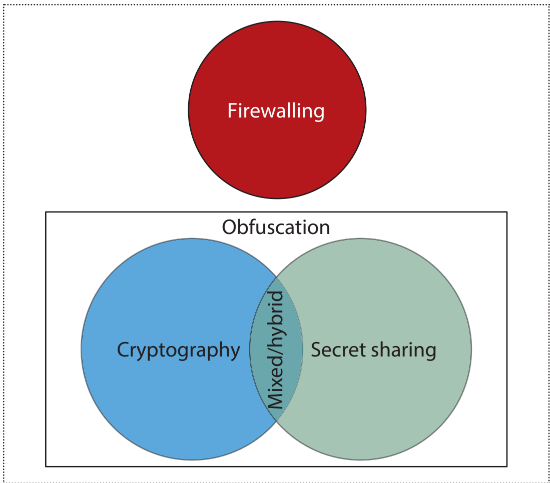


Figure 1. Classes of confidentiality-preserving schemes.

Firewall-Based Systems

Firewall-based confidentiality-preserving BFT systems assume that a system operates under state-machine replication and that all messages have to pass through a filter that is placed between clients and servers. The filter's role is to discard any content that doesn't conform to the majority of a Byzantine-tolerant quorum. In particular, when servers communicate with clients, the filter makes sure that only messages that match the BFT majority's output are allowed to pass through. Nonmatching messages are discarded immediately to prevent leakage.

Jian Yin and his colleagues presented the only general-purpose system based on firewalling that we are aware of.[2] Their system is based on a privacy firewall that separates the ordering of requests from their execution. Several filters separate these two layers, eliminating any Byzantine replies and making covert channel leaks impossible (for example, via reply delay manipulation). Byzantine replies are detected using *threshold signatures*, which are quick to reconstruct and verify but computationally expensive to create. Threshold signatures have been previously used in the context of BFT as a bootstrapping mechanism for key management in encryption-based systems and for securing membership changes.

Besides the performance cost of threshold signatures and increased latency caused by the filter layers, there are two significant downsides to privacy firewalls. First, although firewall-based systems support the execution of arbitrary commands, the support is limited to commands that behave deterministically, that is, each request yields exactly the same bit-by-bit reply from each replica. If replies from different replicas differ even in one bit, then the threshold signature validation algorithm will fail at the first filter layer and all replies will be discarded. For example, any service dealing with server-generated timestamps will require either perfect clock synchronization between replicas or compromise for less accurate timestamps. Because the first layer of filters prevents any reply from passing through unless it matches the shared signature (in other words, matches the majority of replies), clients can't see nonmatching answers, even if they could implement their own logic to handle such cases (for example, averaging the timestamps).

Second, the privacy firewall system introduces two new sources of Byzantine nodes. Whereas in traditional BFT systems, each node was responsible for both order and execution and thus only one type of Byzantine node existed, in the privacy firewall system, there might be up to f Byzantine agreement nodes, up to g Byzantine filter nodes, and up to h Byzantine execution nodes. This requires a heavy investment in infrastructure, even if some of these roles can be combined in the same physical node. For example, to tolerate a single fault of each type, the system must have at least 11 processes running in nine nodes. Agreement nodes require a $3f + 1$ quorum, filter nodes require a $(g + 1)^2$ quorum, and execution nodes require a $2h + 1$ quorum. Finally, the execution nodes must be on a separate network partition from the agreement nodes, and they must interface only through the filter nodes, which must also be partitioned in $g + 1$ layers.

Given these constraints, the firewall approach is usually restricted to cases in which cloud providers let customers dictate network topology.

Obfuscation-Based Systems

In the obfuscation class of BFT systems, data leakage is prevented by an obfuscation mechanism. Even if a replica leaks information, that information will be obfuscated and therefore useless. The corollary is that obfuscating systems are only as strong as their obfuscation scheme, which can be cryptographic, secret sharing, or a combination of both.

Obfuscation through Cryptography

The most common approach to obfuscation is through cryptography.[3] In fact, when Byzantine fault tolerance isn't involved, it seems to be the standard answer for confidentiality-preserving systems.

In the case of BFT systems, encryption is usually employed to obfuscate the data, which is then submitted to potentially Byzantine participants for storage. Server-side data manipulation is currently impossible or impractical (for example, it's many orders of magnitude slower than plaintext manipulation), although recent results have shown that it might eventually become practical.[4]

However, encryption doesn't solve the confidentiality problem. It just shifts the problem from preserving the data's confidentiality to preserving the encryption keys' confidentiality. Because encryption keys can't be trusted by servers, key distribution and revocation are nontrivial problems, which in turn generate high management costs. Some systems try to solve the problem of key management by employing secret sharing to store the keys on the Byzantine servers, effectively running a secret sharing–based system to bootstrap the cryptography-based system.[5]

Obfuscation through Secret Sharing

Using secret sharing to preserve confidentiality in BFT systems relies on the idea that a shared secret can be reconstructed only if a sufficient number of shares is provided.[6] By making the minimum number of shares required to reconstruct the secret larger than the expected number of Byzantine members, information leaked by those members can't lead to the secret's reconstruction.

The main disadvantage of these systems is that all confidential data going to or coming from the replicas has to pass through the secret-sharing scheme. Because each server possesses a different share and clients need to be able to validate each share, their functionality as simple storage systems is limited. Due to secret-sharing algorithms' assumed poor performance, most of the literature focuses on improving the performance rather than making more general-purpose secret-sharing systems. The concern about performance comes mainly from the need to validate the obfuscated data, which is an area of intensive research[7] and is usually done by introducing a token with the data, such as a checksum, a digest, or a previously agreed-upon piece of information. By introducing this token, the secret-sharing scheme's properties are lost.

The cost of confidentiality is becoming a fair tradeoff in the cloud's privacy-hostile environment.

A fully homomorphic secret-sharing scheme supports noninteractive execution of both additive and multiplicative operations. This means that both addition and multiplication can be performed directly on the shares without the need to interact with other replicas. Such a scheme would make possible the implementation of secure multiparty computation on the server side—in other words, a system capable of implementing arbitrary server-side operations while preserving confidentiality.

Unfortunately, no scheme currently allows full noninteractive homomorphism. Adi Shamir's secret-sharing scheme, for example, has been proven to support additive homomorphism.[8] It's possible to transform Shamir's scheme into a fully homomorphic scheme by making it interactive—multiplications need an extra round of communication between replicas. In our 2011 paper "Belisarius: BFT Storage with Confidentiality," we proposed a BFT storage system that exploits the homomorphic property of Shamir's scheme to enable some server-side operations.[9]

Preserving confidentiality in the cloud comes at a cost. First, the application of BFT techniques requires more replicas than traditional fault-tolerance techniques and causes higher latencies. When firewalling is employed, the number of servers increases, as does the complexity of the network topology. Obfuscation by cryptography adds a layer of indirection because the system's weakness becomes the management and confidentiality of the encryption keys. Obfuscation by secret sharing requires isolation of replicas and impacts performance. However, the cost of confidentiality is becoming a fair tradeoff in the cloud's privacy-hostile environment. ■

References

1. L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Trans. Programming Lan-*

guages and Systems*, vol. 4, no. 3, 1982, pp. 382–401.
2. J. Yin et al., "Separating Agreement from Execution for Byzantine Fault Tolerant Services," *Proc. 19th ACM Symp. Operating Systems Principles* (SOSP 03), 2003, pp. 253–267.
3. E. Damiani et al., "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs," *Proc. 10th ACM Conf. Computer and Communications Security* (CCS 03), 2003, pp. 93–102.
4. C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," *Proc. 41st Ann. ACM Symp. Theory of Computing*, 2009, pp. 169–178.
5. M.K. Reiter et al., "The Omega Key Management Service," *Proc. 3rd ACM Conf. Computer and Communications Security* (CCS 96), 1996, pp. 38–47.
6. D.R. Stinson, "An Explication of Secret Sharing Schemes," *Designs, Codes and Cryptography*, vol. 2, no. 4, 1992, pp. 357–390.
7. C.-C. Chang and C.-W. Chan, "Detecting Dealer Cheating in Secret Sharing Systems," *Proc. 24th Ann. Int'l Computer Software and Applications Conf.* (COMPSAC 00), 2000, pp. 449–453.
8. A. Shamir, "How to Share a Secret," *Comm. ACM*, vol. 22, no. 11, 1979, pp. 612–613.
9. R. Padilha and F. Pedone, "Belisarius: BFT Storage with Confidentiality," *Proc. 10th IEEE Int'l Symp. Network Computing and Applications*, 2011, pp. 9–16.

Ricardo Padilha is a research assistant at the University of Lugano, Switzerland. Contact him at ricardospadilha@gmail.com.

Fernando Pedone is a professor at the University of Lugano, Switzerland. Contact him at fernando.pedone@usi.ch.

This article originally appeared in IEEE Security & Privacy, *vol. 13, no. 1, 2015.*

Editor: **Christof Ebert**
Vector Consulting Services
christof.ebert@vector.com

Infrastructure as a Service and Cloud Technologies

Nicolás Serrano, Gorka Gallardo, and Josune Hernantes

Software and infrastructure are increasingly consumed from the cloud. This is more flexible and much cheaper than deploying your own infrastructure, especially for smaller organizations. But the cloud obviously bears a lot of risk, as our partner magazine *Computer* emphasized in its October 2014 issue. Performance, reliability, and security are just a few issues to carefully consider before you use the cloud. The perceived savings immediately turn into a large extra cost if your clients and workforce are disconnected for some minutes or critical infrastructure doesn't perform as intended—which happens more often than what brokers and providers disclose. In this issue's column, IT expert Nicolas Serrano and his colleagues examine the enterprise cloud market and technologies. They provide hands-on guidance for making the right decisions. I look forward to hearing from both readers and prospective authors about this column and the technologies you want to know more about. —*Christof Ebert*

CLOUD COMPUTING'S LOW COST, flexibility, and agility are well understood in today's corporate environment. However, to fully exploit cloud technologies, you need to understand their best practices, main players, and limitations.

The concept of cloud computing has existed for 50 years, since the beginning of the Internet.[1] John McCarthy devised the idea of time-sharing in computers as a utility in 1957. Since then, the concept's name has undergone several changes: from service bureau, to application service provider, to the Internet as a service, to cloud computing, and to software-defined datacenters, with each name having different nuances. However, the core concept is the same: providing IT services based on the Internet (the cloud).

The most-used definition of cloud computing belongs to the US National Institute of Standards and Technology (NIST):

> *Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.*[2]

Providers use three well known models (see Figure 1): IaaS (infrastructure as a service), PaaS (platform as a service), and SaaS (software as a service). Here, we focus on IaaS. The next step is to decide on a model for deploying cloud services. In a *public cloud*, a provider provides the infrastructure to any customer. A *private cloud* is offered only to one organization. In a *hybrid cloud*, a company uses a combination of public and private clouds.

To choose the most appropriate cloud-computing model for your organization, you must analyze your IT infrastructure, usage, and needs. To help with this, we present here a picture of cloud computing's current status.

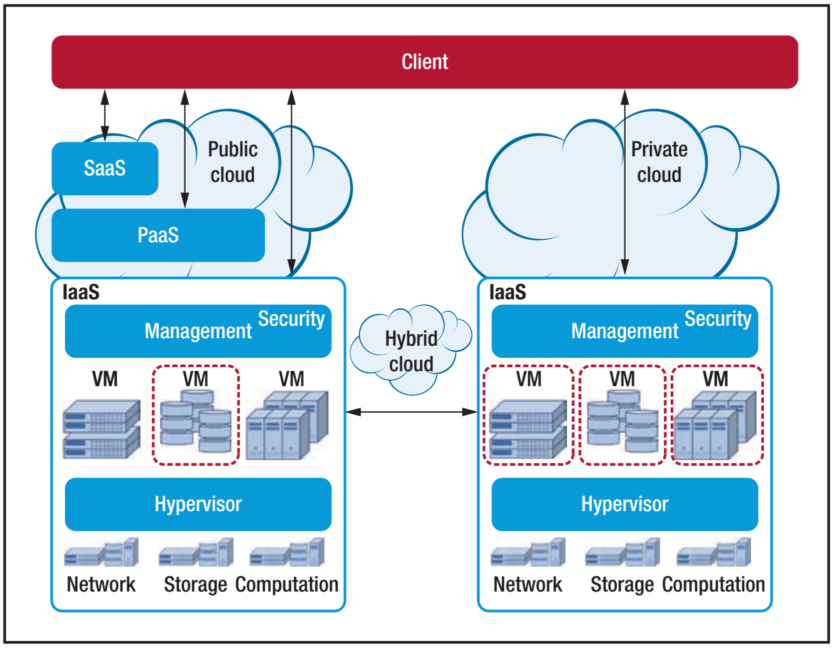


FIGURE 1. The three cloud models. IaaS is infrastructure as a service, SaaS is software as a service, PaaS is platform as a service, and VM stands for a virtual machine.

Cloud Computing Best Practices

As with every new architectural paradigm, it's important to design your systems taking into account the new technology's characteristics. To select a cloud provider or technology, you should understand your requirements in order to list the needed features. Here are some best practices for cloud migration.[3]

An Elastic Architecture

IaaS offers precise scalability. The cloud can outperform physical hardware's classic scale-up or scale-out strategies. To gain as much as you can from this potential, architect your systems and application with as much decoupling as possible, using a service-oriented architecture and using queues between services.

Design for Failure

High scalability has limitations. IaaS technology and architecture lead to a less robust system because you're replacing hardware with several software layers, adding obvious complexity and failure points. Redundancy and fault tolerance are primary design goals.

Besides having an established backup strategy, to assure business continuity, ensure your system is prepared for reboots and relaunches. Automation in your deployment practice is a must, with recipes for server configuration and deployment. Providing automation requires new development practices (development and operations management, continuous integration, test-driven development, and so on) and new tools such as Chef, Puppet, or Ansible.

High Availability

IT resource disruption has a huge negative impact on any business. Lost control of the underlying infrastructure when moving to the cloud, and the fact than the service-level agreement (SLA) won't cover all the incurred costs, should lead you to design with outages and high availability in mind. With the ease of creating virtual instances, deploying clusters of servers or services is a popular approach. In this scenario, load balancing is a well-established technique for operating with clusters; it's an important feature to consider when selecting a cloud provider.

It's also important to use several available zones or at least different datacenters to make your system as robust as possible. Amazon Web Services (AWS) experienced this in April 2011 when its systems didn't run or ran intermittently for four days. Separating clusters into regions and datacenters will increase your resources' resilience.

Performance

You need to consider the technology's limitations regarding performance—

TABLE 1

Cloud provider characteristics.

Company	Reported usage (%)*[4]	Trend 2014–2015 (%)*[4]		Technology	Price (US$, Aug. 2013)[5]	Functionality or features[6]					Datacenters	Vendor positioning[7]
		Experimenting	Plan to use			Advanced features or functionality	Hybrid cloud	Network management	Service-level agreement (%)[4]	Average service time (yrs.)		
Amazon Web Services	54	23	9	Proprietary	66	Good	Yes	Partial	99.95	5+	US East and West Coast, Ireland, Japan, Singapore, Australia, Brazil, China	Leader
Microsoft Azure	6	20	8	Proprietary	65	Good	Yes	Partial (unavailable for Linux)	99.95	4	US, Ireland, Netherlands, Hong Kong, Japan, Singapore, China, Brazil	Leader
Rackspace	12	14	12	OpenStack	116	Adequate	Yes	No	100	5+	Central and eastern US, UK, Australia, Hong Kong	Niche player
HP	4	8	8	OpenStack	87	Good	Yes	No	99.95	2	Eastern and western US	Niche player
IBM	4	9	7	OpenStack	182	Good	Yes	No	100	5+	US, Netherlands, Singapore	Niche player
Google	4	17	13	Proprietary	42	Adequate	Yes	No	99.95	2	Central US, Belgium, Taiwan	Visionary
Others†	24	45	31	—	—	—	—	—	—	—	—	—

** In these columns, the sum of the percentages is greater than 100 because some companies use several products.*
† This group includes PaaS (platform as a service) and recent providers.

mainly, lack of isolation and lost robustness. In any multitenant environment, an instance's performance can be affected by your neighbors. A usage burst in a neighbor's instance can affect the available resources, notably compute units and disks' IOPS (I/O operations per second). Your architecture should deal with these changes.

Also, bottlenecks might arise owing to latency issues, even within instances at the same datacenter. Cloud providers offer some features to deal with this (for example, AWS placement groups). However, if your architecture has servers at different regional datacenters, you should use other techniques (for example, caching).

Security

Because of a public cloud's open characteristics, designing and maintaining a secure infrastructure should be an important driver in any cloud deployment. Enforce well-established security practices: firewalls, minimal server services to reduce attack vectors, up-to-date operating systems, key-based authentication, and so on. But challenges might arise from the increased number of servers to maintain and the use of the cloud for different development environments: development, staging, and production. In this scenario, isolating and securing each environment is important because a breach in a prototyping server can give access through the secret keys to the whole infrastructure.

Monitoring

The ease of deploying new resources can make the number of servers grow exponentially. This raises new issues, and monitoring tools are vital to system management. First, they play a basic role in automatic scaling on a cyclical basis and based on events. Second, they're part of the tools needed to ensure a robust architecture, as the Netflix Chaos Monkey showed. Finally, they're important for detecting security breaches and forensic investigation, as some security breaches have shown.

Public Clouds

The public cloud was the first type of cloud offered to the general public, when AWS offered its experience with its private cloud to the general public. When you're selecting a vendor, it's important to consider several factors, mainly cost, performance, features, data location, and availability. But because the public cloud is a fairly recent technology, you should also consider vendor positioning and future use trends (see Table 1).

Cloud providers are battling for market position, which is leading them to reduce their public IaaS cloud prices, offering attractive solutions.

It's important to select the most effective vendor from a performance–cost perspective. However, your comparison should also consider whether the performance level is guaranteed, startup times, scalability responsiveness, and latency. These factors might vary among providers and impact the infrastructure's responsiveness.

The datacenters' location can affect your decision. The provider should comply with data privacy laws and corporate policies; the server locations should be based on these considerations. These restrictions might vary among countries and among companies. You might find you'll need to have all data under the same jurisdiction (for example, in Europe). In other cases, Safe Harbor principles, in which US companies comply with EU laws, can be good enough.

Understanding each player's SLA is important. But because almost every provider offers high-enough service levels (more than 99.95 percent), it's important to evaluate the accountability the SLA offers in case of noncompliance. Normally, this won't cover the costs of outages, so your infrastructure should be prepared for them.

Providers

Once you've defined your selection criteria, you can compare providers. The following are the most relevant ones.

Amazon. AWS (http://aws.amazon.com) continues being the dominant player in cloud computing, thanks to Amazon having been the first company to offer cloud services, in 2006.

AWS is cost effective. Its pay-as-you-go model lets you scale cloud capacity up or down without paying a high price. It also offers many additional IaaS services and integrated monitoring tools. It's particularly valuable for startups and agile projects requiring quick, cheap processing and storage.

Because AWS is a general provider, you can operate independently, which is convenient for normal operations but becomes risky when problems occur. Extensive technical support is a premium feature, whereas most of AWS's competitors offer it as a standard feature.

Microsoft Azure. Azure (http://azure.microsoft.com/en-us) entered the cloud IaaS market in February 2010. It has a large market share and is a good candidate because of its market position in other areas. It offers compute and storage services similar to those of other IaaS providers, and it allows full control and management of virtual machines. Additionally, Azure's UI is easy to use, especially for Windows administrators. However, because the Azure offering is newer than Amazon's or Rackspace's, it still has many features in "preview" mode and still has networking and security gaps.

Rackspace. Rackspace (www.rackspace.com) is a founder of OpenStack (which we describe later) and a major player in open source cloud IaaS. It hosts more than half of the Fortune 500 companies at its datacenters, while strongly focusing on SMEs (small-to-medium enterprises).

Rackspace provides an inexpensive, intuitive cloud with optional managed services and an easy-to-use control panel that suit SMEs. It also guarantees extensive support. However, it has limited pricing options, providing only month-to-month subscriptions. Also, it doesn't offer specialized services.

Google. Although Google AppEngine was a pioneer of cloud computing in the PaaS model, Google Compute Engine (https://cloud.google.com/compute) is relatively new

TABLE 2 The main products used to create private clouds.*

Solution	Reported usage (%)[4]	License	Trend, 2013–2014[4]
VMware	43	Proprietary	Down
OpenStack	12	Open source	Up
Microsoft	11	Proprietary	Down
CloudStack	6	Open source	Down
Eucalyptus	3	Open source	Down

** Some of the included solutions, as stated in* State of the Cloud Report,[4] *don't strictly meet all cloud-computing requirements.*

to the IaaS market. Nevertheless, Google's number of physical servers and global infrastructure make it a good candidate. Moreover, Google Compute Engine is well integrated with other Google services such as Google Cloud SQL and Google Cloud Storage.

Google Compute Engine is well suited for big data, data warehousing, high-performance computing, and other analytics-focused applications. Its main limitation is that it doesn't integrate administrative features. So, users must download extra packages.

HP. HP is still relatively new in the IaaS game; it launched its service in December 2012. Its public cloud, HP Cloud Compute (www.hpcloud.com), is built on OpenStack and offers a range of cloud-related products and services. It's a good candidate owing to its positioning in the server market. Its IaaS offering supports public, hybrid, and private clouds.

HP Cloud Compute is a good solution for companies that want to integrate their existing IT infrastructure with public-cloud services and invest in a hybrid cloud.

IBM. IBM's resources, size, and knowledge of datacenters make it another player to consider. IBM Cloud (www.ibm.com/cloud-computing/us/en) offers core computing and storage services. This IaaS is best for large enterprises with heavy data-processing needs and security concerns.

IBM Cloud provides a good combination of management, software, and security features for administrators. However, its focus is limited to medium-to-large enterprises and enterprises whose main provider is IBM.

Issues and Concerns

When considering adoption of a cloud architecture, it's important to understand what the technology can offer you and the main issues you'll have to deal with in each of these new infrastructures. Only by clearly understanding each of the approaches' business and technical opportunities and limitations will you be able to select the best option on the basis of your needs.

Besides the economic advantages from a cost perspective, the main competitive advantages are the flexibility and speed the cloud architecture can add to your IT environment. In particular, this kind of architecture can provide faster deployment of and access to IT resources, and fine-grain scalability.

A recent survey indicated the issues that beginner and experienced enterprise cloud users face.[4] For beginners, the main issues are security, managing multiple clouds, integration with current systems, governance, and lack of expertise. Experienced companies face issues of compliance, cost, performance, managing multiple clouds, and security.

The differences are understandable. Different problems arise on the basis of the degree of advancement of cloud architecture adoption. Early on, the main issues are resource expertise and control because the company hasn't acquired enough knowledge of and experience with the architecture. For more experienced companies, performance and cost are important because the architecture's limitations might have started emerging.

Both groups must deal with security, compliance, and managing multiple clouds. Regarding security and compliance, some problems might arise from the multitenant architecture. Some of these problems might not be solved, which might tip the balance toward a private or hybrid cloud. Such a decision is plausible, in keeping with the issue of managing multiple clouds.

Private and Hybrid Clouds

To solve the issues with public clouds, cloud-computing providers introduced the private cloud. This cloud might be in the organization's buildings, in the farm of the organization's provider, or in another provider's datacenter. Usually, it will be virtualized, but other combinations are possible. The important element is that only the customer's organization can operate it.

Because all private-cloud products allow integration with public clouds, we discuss both private and hybrid clouds here. Table 2 shows the main products used to create private clouds.

Eucalyptus

Eucalyptus (www.eucalyptus.com) released its first product in 2008. Nowadays the company provides its software as open source products and services. (Recently, Eucalyptus was bought by HP, a supporter of OpenStack.) From the company's download area, you can install a private cloud on your computer. From its product area, you can contract servers for your private cloud.

Eucalyptus software's main advantage is its AWS compatibility (see Table 3), based on a partnership with Amazon. So, some features that AWS makes available for the public cloud are applicable to Eucalyptus services.

Eucalyptus software's weak points are the limited GUI and the risk of uncertainty generated by AWS's private-cloud strategy: AWS offers Amazon Virtual Private Cloud and a connection to a hardware VPN (virtual private network).

TABLE 3. Eucalyptus compatibility with Amazon Web Services (AWS).

AWS services	Eucalyptus components
Amazon Elastic Compute Cloud (EC2)	Cloud Controller
Amazon Elastic Block Storage (EBS)	Storage Controller
Amazon Machine Image (AMI)	Eucalyptus Machine Image
Amazon Simple Storage Service (S3)	Walrus Storage
Amazon Identity and Access Management (IAM)	IAM
Auto Scaling	Auto Scaling
Elastic Load Balancing (ELB)	ELB
Amazon CloudWatch	CloudWatch

OpenStack

OpenStack (www.openstack.org) is the other main player in the private-cloud field. It's also open source, and its greatest strength is its support from companies such as AT&T, AMD, Cisco, Dell, HP, IBM, Intel, NEC, Red Hat, VMware, and Yahoo.

OpenStack is complex, with different components and multiple command-line interfaces. Competitors say it's not a product but a technology. This can be a barrier for nontechnical companies but not for public- and private-cloud providers, which are OpenStack's main users. For them, an open source product is attractive because, just as with using Linux in server computers, there are cost and portability advantages for the end user.

Portability is another important feature of OpenStack because end users don't want to be locked into a particular provider. However, providing the option of portability can be an issue for providers that want to offer differentiated proprietary features.

CloudStack

Citrix purchased CloudStack (http://cloudstack.apache.org) from Cloud.com. Citrix donated it to the Apache Software Foundation, which released it after it spent time in the Apache Incubator.

Unlike OpenStack, CloudStack offers a complete GUI and a monolithic architecture that simplifies installing and managing the product. Like OpenStack, most installations belong to service providers. CloudStack also offers AWS compatibility through an API translator.

Proprietary Solutions

VMware (www.vmware.com) and Microsoft (www.microsoft.com/enterprise/microsoftcloud) emphasize the hybrid nature of their offerings. They have products for both public and private clouds and provide on-premises servers.

VMware products include vCloud Hybrid Service, vCloud Connector, and vSphere virtualization. Microsoft has Windows Azure, Windows Server, and Microsoft System Center.

These two providers offer a more integrated solution because they own their products, but the disadvantage is lack of portability.

The public-cloud market has some years of history and well-known players. But remember that the cloud-computing market is growing. Newcomers are always entering, and the leaders in public- and private-cloud services can change.

So, your selection of a cloud-computing model and provider must take into account the factors listed in Tables 1 and 2, a service's specific purpose, and the elements of the application you want to migrate to the cloud. The approach and reach of your cloud adoption efforts will be limited by each situation. For example, your application architecture and the technology involved won't be the same if you're migrating an application not yet developed or an existing legacy system.

Regarding a new application,

This article originally appeared in IEEE Software*, vol. 32, no. 2, 2015.*

BUNTPLANET

BuntPlanet (http://buntplanet.com) is a small-to-medium enterprise focusing on software engineering. It was founded in 2000 in San Sebastian, Spain. It offers a range of applications mainly for utilities, and it develops custom applications using agile practices.

Although BuntPlanet had some experience with cloud services, it was only in 2009 that it decided to use the cloud for its internal applications. The availability of Amazon's European datacenter was a determining factor because BuntPlanet could comply with Spanish data protection laws. During the selection process, the company compared other alternatives but chose Amazon mainly for the extensive features and competitive price.

At first, BuntPlanet replicated its server architecture in the cloud with Amazon EC2 (Elastic Compute Cloud) servers. Since then, it has incrementally refactored its applications to realize the full potential of a cloud architecture. It now uses other Amazon services such as S3 (Simple Storage Service) for data storage, Glacier for backups, RDS (Relational Database Service) for relational databases, SNS (Simple Notification Service) for push notifications, and SQS (Simple Queue Service) for queues.

To achieve the required service level, BuntPlanet uses reserved instances and a second redundant region in Europe. It's also self-monitoring its cloud resources and has set alerts for failures. This approach lets the company achieve its desired reliability and availability.

BuntPlanet's experience with the public cloud has been positive from economical and technical viewpoints. Using a public cloud has allowed BuntPlanet to simplify processes and minimize the need to support a big hardware infrastructure and a high bandwidth in its installations. As a result, it's promoting this architecture in its customer projects, using the public cloud for load tests and production environments.

you should develop it with an elastic architecture and best practices in mind. Decouple the presentation, business, and logic layers in several services and use a queue system to communicate between them. A high number of servers, a fault-tolerant design, and automatic provisioning will require high-level features from the cloud provider or technology.

Regarding a complete legacy system, refactoring the application to achieve decoupling isn't feasible. A pure cloud architecture is impossible, and a reduced list of features is required. Your priority should be virtual instances' robustness and reliability.

Other scenarios, such as disaster recovery or using the cloud when the demand spikes (cloud bursting), require specific cloud technology features.

If you're dealing with a new application and provider independence is a priority, you might prefer an OpenStack provider. If you're migrating a legacy system and you have IT experience with VMware, you might select VMware for your cloud. Regarding cloud bursting in a Microsoft Server IT environment, you might choose the Microsoft solution. However, AWS, a market leader and proven feature-rich platform, is always an option.

As you can see, because of the variety of choices, different customers might choose different platforms. For example, HP and Rackspace (service providers), Cybercom (a consulting company), and eBay (an end user) use OpenStack, whereas VMware and Microsoft customers use their provider's solution. For a look at how one company (BuntPlanet) chose its cloud provider, see the sidebar.

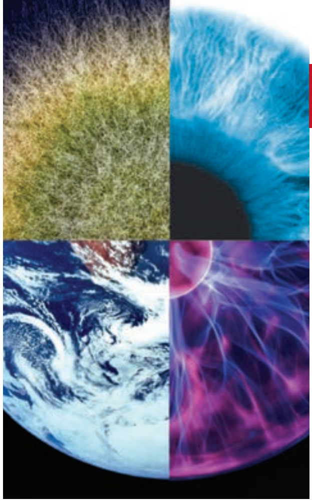
References

1. M. Vouk, "Cloud Computing—Issues, Research and Implementations," *J. Computing and Information Technology*, vol. 16, no. 4, 2008, pp. 235–246.
2. P. Mell and T. Grance, *The NIST Definition of Cloud Computing*, US Nat'l Inst. of Standards and Technology, 2011.
3. F. Fehling, F. Leymann, and R. Retter, "Your Coffee Shop Uses Cloud Computing," *IEEE Internet Computing*, vol. 18, no. 5, 2014, pp. 52–59.
4. *State of the Cloud Report*, RightScale, 2014; www.rightscale.com/lp/2014-state-of-the-cloud-report.
5. T. Rodrigues, "Top Cloud IaaS Providers Compared," *Enterprise Cloud*, 27 Aug. 2013; www.techrepublic.com/blog/the-enterprise-cloud/top-cloud-iaas-providers-compared.
6. "Vendor Landscape: Cloud Infrastructure-as-a-Service," Info-Tech Research Group, 2014; www.infotech.com/research/ss/select-the-right-cloud-infrastructure-server-partner.
7. *Magic Quadrant for Cloud Infrastructure as a Service*, Gartner, May 2014; www.gartner.com/technology/reprints.do?id=1-1UKQQA6&ct=140528&st=sb.

NICOLÁS SERRANO is a professor of computer science and software engineering at the University of Navarra. Contact him at nserrano@tecnun.es.

GORKA GALLARDO is a professor of information systems at the University of Navarra. Contact him at ggallardo@tecnun.es.

JOSUNE HERNANTES is a professor of computer science and software engineering at the University of Navarra. Contact her at jhernantes@tecnun.es.



Editor: **Elena Simperl,** University of Southampton, e.simperl@soton.ac.uk

Human Beyond the Machine: Challenges and Opportunities of Microtask Crowdsourcing

Ujwal Gadiraju, *L3S Research Center*
Gianluca Demartini, *University of Sheffield*
Ricardo Kawase and Stefan Dietze, *L3S Research Center*

The notion of *crowdsourcing* existed long before the term itself was coined. The idea that small contributions from a group of individuals can be accumulated to accomplish some work or attain an objective has been observed in different realms of daily life for a few centuries. Over the years, crowdsourcing has emerged as a useful paradigm, showcasing the adage that "the whole is greater than the sum of its parts."

The Web's emergence as a sociotechnical system has dramatically changed the scale and scope of crowdsourcing, opening the possibility of reaching crowds at an unprecedented global scale. Crowdsourcing has proven to be an increasingly important source of knowledge and data, as documented by prominent examples such as Wikipedia, where many authors contribute discrete and diverse information to form an authoritative reference knowledge base. The recent Wikidata project (www.wikidata.org) and the popular ReCAPTCHA work by Luis von Ahn and colleagues[1] are exemplary. Moreover, a recent report by eYeka (http://tinyurl.com/mqsrets) on the state of crowdsourcing in 2015 shows high adoption in business contexts, documenting that 85 percent of the top global brands (as defined at www.bestglobalbrands.com) use crowdsourcing for various purposes.

Particularly in research and science, crowdsourcing has found noteworthy applications in solving real-world problems, ranging from intricate tasks such as protein folding and biomolecule design[2] and mapping outer space[3] to aiding disaster relief initiatives[4] and assembling dictionaries.[5]

With the ubiquity of the Internet, and the concomitant accessibility of established microtask crowdsourcing platforms such as Amazon's Mechanical Turk (MTurk; www.mturk.com) and, more recently, CrowdFlower (www.crowdflower.com), researchers and practitioners are actively turning toward paid crowdsourcing to solve data-centric tasks that require human input. Popular applications include building ground truths, validating results, and curating data. These developments make it possible to build novel intelligent systems that combine the scalability of machines over large amounts of data with the power of human intelligence to solve complex tasks, such as audio transcription, language translation, and annotation. For the rest of this article, *crowdsourcing* will refer to *paid microtask crowdsourcing* (wherein crowd workers receive monetary compensation for successfully completing a microtask). For a thorough discussion of related terms and tasks in the context of human computation and collective intelligence, see the work of Alexander Quinn and Benjamin Bederson.[6]

Owing to the diversity in the crowd in terms of workers' motivations, demographics, and competencies, both microtask design and quality control mechanisms play an unparalleled role in determining the effectiveness of crowdsourcing systems.[7] These two realms, which specifically are concerned with the requesters' perspective on microtasks, have thereby attracted much interest and are our focus in this article.

We summarize and discuss findings from some previous work related to microtask performance and design.[8–10] Although crowdsourcing presents many open challenges, including ethical concerns, we focus on performance-related aspects. In this article, we provide an overview of frequently crowdsourced microtasks, malicious activity observed in crowds, and open research challenges in the field.

Overview of Frequently Crowdsourced Microtasks

To further the understanding of crowdsourced tasks on popular crowdsourcing platforms such as MTurk and CrowdFlower, we categorized typically crowdsourced tasks into a two-level taxonomy,[8] using responses from 567 crowd workers regarding their previously completed tasks. The top level comprises goal-oriented classes, and the second level describes the workflow that a worker must adopt to complete a microtask. We describe the top-level classes below:

- *Information finding* (IF) tasks delegate the process of searching to satisfy an informational need to the workers in the crowd. (For example, "Find information about a company in the UK.")
- *Verification and validation* (VV) tasks require workers in the crowd to either verify certain aspects as per the given instructions or confirm the validity of various kinds of content. (For example, "Is this a spam bot? Check whether the Twitter users are either real people or organizations or spam user profiles.")
- *Interpretation and analysis* (IA) tasks rely on the crowd workers to use their interpretation skills during task completion. (For example, "Choose the most suitable category for each URL.")
- *Content creation* (CC) tasks usually require workers to generate new content for a document or website. They include authoring product descriptions or producing question-answer pairs. (For example, "Suggest names for a new product.")
- *Surveys* (S) about a multitude of aspects ranging from demographics to customer satisfaction are crowdsourced. (For example, "Mother's Day and Father's Day survey [18- to 29-year-olds only!]")
- *Content access* (CA) tasks require the crowd workers to simply access some content. (For example, "Click on the link and watch the video," or "Read the information by following the website link." In these tasks, the workers are asked to consume some content by accessing it, but to do nothing further.)

Note that, depending on the final goal of the task, tasks with the same workflow can belong to different classes at the top level of the microtask taxonomy.

Microtask Evolution over the Years

The most widely used microtask crowdsourcing platform for academic purposes is Mechanical Turk. There has been a constant evolution of its usage patterns since it was launched in 2005. We presented a large-scale analysis of log data from this microtask crowdsourcing platform,[9] showing how the use of microtask crowdsourcing has evolved over the past five years.

Analyzing data from www.mturk-tracker.com about 130 million tasks posted on MTurk between 2009 and 2014, we observed the evolution of platform usage in terms of task type, task pricing, work volume, and required workers. Our main findings are as follows:

- Task reward has increased over time, reaching \$0.05 as the most popular reward level in 2014.
- Tasks requesting audio transcriptions are now the most popular tasks in the platform.
- In terms of task type, CA tasks have become less popular over time, while surveys are becoming more popular, especially for workers based in the US.
- Most tasks do not specify a requirement on where the crowd worker is physically located.
- The number of active requesters (that is, those who publish tasks to be completed) has increased over time, with a rate of 1,000 new requesters per month over the past two years.
- The overall amount of tasks being published and completed on the platform is considerable: 10,000 new tasks are published and 7,500 tasks are completed per hour. Certain requesters can obtain a work throughput of up to thousands of tasks completed per minute.
- Newly published tasks are almost 10 times more attractive for workers compared to old tasks.

Our findings indicate that requesters should engage with workers and publish large volumes of HITs to more quickly obtain data back from the crowdsourcing platform.

Breaking Bad: Typology of Workers and Malicious Activity

Our findings clearly indicate rapid growth in microtask crowdsourcing. Given the crowd workers' inherent characteristics, with respect to their demographics, diversity, and motives, quality control mechanisms that can make crowdsourcing processes more effective play a crucial role. Previous work has asserted that crowdsourced microtasks are often hindered by the presence of spammers and malicious workers who aim to complete microtasks as quickly as possible to gain

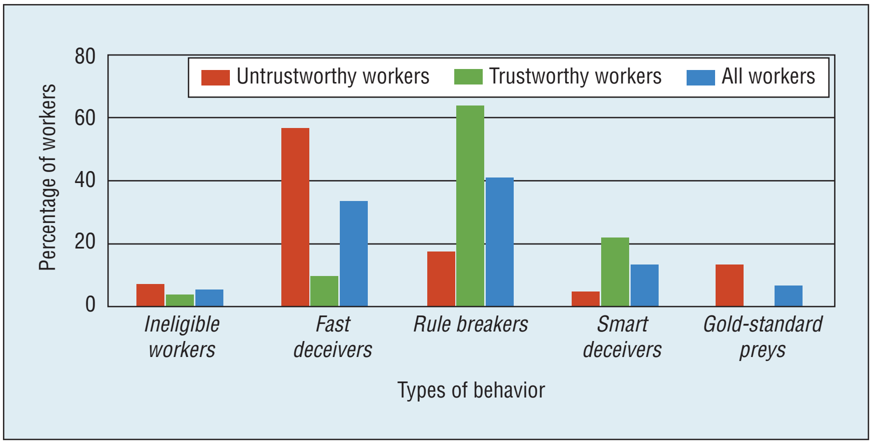


Figure 1. Distribution of trustworthy and untrustworthy workers as per the behavioral patterns exhibited.[10]

monetary rewards.[11,12] We delved into worker behaviors that determine performance, and therefore the quality, of crowdsourced results.[10] Crowd workers exhibit distinct behaviors that affect the overall quality of the work. We hypothesize that by understanding the behavioral patterns of trustworthy workers (workers who pass gold-standard test questions) and untrustworthy workers (workers who fail one or more gold-standard test questions), requesters can inhibit malicious workers from adversely affecting the task output.

On the basis of the responses we received from 1,000 workers in a crowdsourced survey task, we determined the following behavioral typology of trustworthy and untrustworthy crowd workers:[10]

- *Ineligible workers* (IE). Microtask requesters present instructions to the workers that they should follow to complete a given task successfully. The workers who do not qualify as per previously stated requisites belong to this category.
- *Fast deceivers* (FD). Malicious workers are characterized by behavior that suggests a zeal to earn quick money by exploiting microtasks. This is apparent from some workers who adopt the "fastest-response-first" approach, such as copy-pasting the same response in multiple fields.
- *Smart deceivers* (SD). Some eligible workers who are malicious attempt to deceive task administrators by cleverly adhering to the rules. Such workers mask their real objective by simply not violating or triggering implicit validators.
- *Rule breakers* (RB). A behavior prevalent among malicious workers is the lack of conformation to clear instructions with respect to each response. Data collected as a result of such behavior has little value, because the resulting responses might not be useful to the extent intended by the task administrator.
- *Gold-standard preys* (GSP). Some workers who abide by the instructions and provide valid responses surprisingly fall short at the gold-standard questions. They exhibit nonmalicious behavior but stumble at one or more of the gold-standard test questions because of inattentiveness, fatigue, or boredom.

Figure 1 depicts the distribution of the workers whose responses were studied and consequently classified, as per the behavioral pattern they exhibited. It is interesting to note that fast deceivers are the most prevalent type of untrustworthy workers in crowdsourced surveys.

In addition, we found a strong correlation between a worker's task completion time and the malicious intent exhibited. We also introduced a worker's tipping point as an early indicator of possible malicious activity, which can help us detect and reduce malicious activity and hence improve worker performance. For more information, see our previous work.[10]

To restrict the participation of ineligible workers, task administrators should employ prescreening methods. Stringent validators should be used to ensure that fast deceivers cannot bypass open-ended questions by copy-pasting identical or irrelevant material as responses. Rule breakers can be curtailed by ensuring that basic response-validators are employed, so that workers cannot pass off inaccurate or nearly fair responses. Lexical validators can enforce workers to meet the task's exact requirements and prevent ill-fitting responses. Smart deceivers can be restricted by using psychometric approaches (for instance, repeating or rephrasing the same question periodically and cross-checking whether the respondent provides the same response).

Crowdsourcing: The Right Hammer for Every Nail?

Crowdsourcing is becoming a ubiquitous approach to dealing with machine-based computation's limitations by leveraging human intelligence at scale. However, as we have described, crowdsourcing-based solutions must deal with higher uncertainty with respect to worker performance. To make crowdsourcing approaches more efficient and effective, one must consider various aspects. In this article, we list the open challenges when aiming toward wider adoption of crowdsourcing.

In the absence of adequate measures to control crowd workers' performance, crowdsourced data has been shown to deviate far from the desirable. However, it may not always be possible to enforce constraints through task design. Depending on the task at hand, it is not always straightforward to control the performance in real time. Furthermore, not all tasks can be modeled as microtasks that are fit for crowd workers. Task decomposition to facilitate atomic microtask crowdsourcing, especially in the case of complex tasks, is an interesting open challenge.

In the following, we introduce some of the open research challenges toward making crowdsourcing-based solutions more dependable.

Crowdsourcing efficiency

To make crowdsourcing solutions scale to large amounts of data, it is key to design solutions that will retain crowd workers longer in the crowdsourcing platform[13] and prioritize work execution over the crowd.

Incentives, gamification, and satisfaction. To retain workers on the platform, one can leverage custom task-pricing schemes,[14] gamification techniques,[15] or competitive task designs[16] to recognize worker contributions, direct workers toward specific task types, and balance task difficulty to keep workers productive.

Scheduling tasks. To make sure most important tasks are completed as quickly as possible, scheduling techniques must be considered to prioritize tasks. Such techniques also need to consider worker properties such as training effects, context switch cost, and personal preferences.

Performance monitoring

To obtain higher-quality crowdsourced data, we identify the need to create solutions that better profile crowd workers. This will let us understand which worker will perform with high quality on which tasks. This can be done by automatically detecting malicious workers and effectively routing tasks to the right workers in the crowd.

Detecting poorly performing workers. It is important to effectively detect low-quality workers in the crowd to remove their answers from the generated annotations. This can be done using advanced result aggregation techniques or supervised machine learning approaches.

Task routing. By profiling crowd workers over time, we can understand each worker's strengths and weaknesses. We can then leverage such information to assign microtasks to the right workers in the crowd rather than randomly assigning them.

Crowdsourcing ethics

Crowdsourcing is used in production for many commercial products. For a sustainable crowdsourcing environment, a crowdsourcing ethics culture is required that includes schemes to identify fair pricing of work and build long-term worker perspectives and careers.

Fair pricing. In terms of fair pricing, it's important to design methods to correctly price microtasks rather than relying exclusively on market dynamics (that is, balance of demand and offer). Estimating the effort that a microtask takes to complete is key,[17] but it also requires personalized estimations for different types of workers.

Long-term work perspective. From a long-term work perspective, crowd workers are satisfied of their rewards because of the short-term benefits they perceive. However, in the long term, there is no personal development program or social security scheme in place.[18] A better regulation of the crowdsourcing market is necessary—one that considers crowd worker careers and strives for improved transparency (such as in Turkopticon[19]).

Given the controversial discussion of such ethical concerns, a wider discourse of such topics is required to improve the performance of both the workers and the requesters in crowdsourcing settings. With efforts underway to address such challenges, we anticipate a growing importance of crowdsourcing-based efforts in data-centric tasks, specifically to complement and validate machine-processed results with human intelligence. ■

References

1. L. von Ahn et al., "Recaptcha: Human-based Character Recognition via Web Security Measures," *Science*, vol. 321, no. 5895, 2008, pp. 1465–1468.
2. B.L. Ranard et al., "Crowdsourcing—Harnessing the Masses to Advance Health and Medicine, a Systematic Review," *J. General Internal Medicine*, vol. 29, no. 1, 2014. pp. 187–203.
3. E.C. McLaughlin, "Image Overload: Help Us Sort It All Out, NASA Requests," CNN, 18 Aug. 2014; http://edition.cnn.com/2014/08/17/tech/nasa-earth-images-help-needed.
4. M. Zook et al., "Volunteered Geographic Information and Crowdsourcing Disaster Relief: A Case Study of the Haitian Earthquake," *World Medical & Health Policy*, vol. 2, no. 2, 2010, pp. 7–33.
5. N. Lanxon, "How the Oxford English Dictionary Started Out Like Wikipedia," *Wired*, blog, 13 Jan. 2011;

www.wired.co.uk/news/archive/2011-01/13/the-oxford-english-wiktionary.
6. A.J. Quinn and B.B. Bederson, "Human Computation: A Survey and Taxonomy of a Growing Field," *Proc. SIGCHI Conf. Human Factors in Computing Systems*, 2011, pp. 1403–1412.
7. P.G. Ipeirotis, F. Provost, and J. Wang, "Quality Management on Amazon Mechanical Turk," *Proc. ACM SIGKDD Workshop on Human Computation*, 2010, pp. 64–67.
8. U. Gadiraju, R. Kawase, and S. Dietze, "A Taxonomy of Microtasks on the Web," *Proc. 25th ACM Conf. Hypertext and Social Media*, 2014, pp. 218–223.
9. D.E. Difallah et al., "The Dynamics of Micro-Task Crowdsourcing—The Case of Amazon MTurk," *Proc. 24th Int'l Conf. World Wide Web*, 2015, pp. 238–247.
10. C. Eickhoff and A.P. de Vries, "Increasing Cheat Robustness of Crowdsourcing Tasks," *Information Retrieval*, vol. 16, no. 2, 2013, pp. 121–137.
11. G. Kazai, J. Kamps, and N. Milic-Frayling, "Worker Types and Personality Traits in Crowdsourcing Relevance Labels," *Proc. 20th ACM Int'l Conf. Information and Knowledge Management*, 2011, pp. 1941–1944.
12. U. Gadiraju et al., "Understanding Malicious Behavior in Crowdsourcing Platforms: The Case of Online Surveys," *Proc. 33rd Ann. ACM Conf. Human Factors in Computing Systems*, 2015, pp. 1631–1640.
13. M. Rokicki et al., "Competitive Game Designs for Improving the Cost Effectiveness of Crowdsourcing," *Proc. 23rd ACM Int'l Conf. Information and Knowledge Management*, 2014, pp. 1469–1478.
14. D.E. Difallah et al., "Scaling-Up the Crowd: Micro-Task Pricing Schemes for Worker Retention and Latency Improvement," *Proc. 2nd AAAI Conf. Human Computation and Crowdsourcing*, 2014, pp. 50–58.
15. K. Siorpaes and E. Simperl, "Human Intelligence in the Process of Semantic Content Creation," *World Wide Web*, vol. 13, 2010, pp. 33–59.
16. M. Rokicki, S. Zerr, and S. Siersdorfer, "Groupsourcing: Team Competition Designs for Crowdsourcing," to be published in *Proc. 24th Int'l World Wide Web Conf.*, 2015.
17. J. Cheng, J. Teevan, and M.S. Bernstein, "Measuring Crowdsourcing Effort with Error-Time Curves," *Proc. 33rd Ann. ACM Conf. Human Factors in Computing Systems*, 2015, pp. 1365–1374.
18. A. Kittur et al., "The Future of Crowd Work," *Proc. Conf. Computer Supported Cooperative Work*, 2013, pp. 1301–1318.
19. L.C. Irani and M.S. Silberman, "Turkopticon: Interrupting Worker Invisibility in Amazon Mechanical Turk," *Proc. SIGCHI Conf. Human Factors in Computing Systems*, 2013, pp. 611–620.

Ujwal Gadiraju is a PhD candidate in the L3S Research Center at the Leibniz Universität Hannover. Contact him at gadiraju@l3s.de.

Gianluca Demartini is a lecturer in data science in the Information School at the University of Sheffield. Contact him at g.demartini@sheffield.ac.uk.

Ricardo Kawase is a data analyst for the trust and safety team at mobile.de. He performed the work for this article as a researcher in the L3S Research Center at the Leibniz Universität Hannover. Contact him at rkawase@team.mobile.de.

Stefan Dietze is a research group leader in the L3S Research Center at the Leibniz Universität Hannover. Contact him at dietze@l3s.de.

This article originally appeared in IEEE Intelligent Systems*, vol. 30, no. 4, 2015.*

High Tech @ Home

Although no one has yet to recreate Rosie from the animated sitcom *The Jetsons*, now more than half a century later there are some recently released and upcoming products that capture aspects of that highly skilled household robot. Recognizing family members; reminding them of tasks and events; following them through the house with music, movies, and games; and cooking a customized, artistic breakfast for everyone are just some of the features these new products boast.

Simplicam

ArcSoft's Simplicam has many of the usual features of a Wi-Fi home security camera. With a 107 degree field of view, night vision, and 720-pixel resolution, this camera provides users with a live view of their home through either a Web or mobile (Android or iOS) app. For a monthly fee, recording services allow users to review up to 21 days of recordings and save up to five hours of clips. Users can be notified with alerts when motion or noise is detected, although this can lead to frequent alerts, for example, when a pet walks by, motion is visible through a window, or curtains are blown by an air vent. Simplicam uses face detection to notify the user only when a human is detected, improving the value of the alerts. With the recent addition of face recognition (currently in beta), users can receive customized alerts based on matching that detects the faces of up to 10 specified people. For example, users can choose to be alerted when their children enter the house or only when a stranger enters the house. The camera retails for $149, with storage plans starting at $4.99 per month. See www.simplicam.com for more information.

Figure 1. PancakeBot 3D printer lets users trace any picture to define the path of the batter.

PancakeBot

For those looking for a home robot to help out in the kitchen, the PancakeBot 3D printer will cook the family custom pancakes for breakfast. The idea for this creative household appliance started when Miguel Valenzuela decided to make a pancake machine out of LEGOs for his two young daughters. Two years and several design iterations later, the latest version of the PancakeBot (no longer built out of LEGOs) was launched through Kickstarter and raised over $460,000. Using the included software, users can trace any picture to define the path of the batter. The batter is dispensed onto the PancakeBot griddle, with compressed air and a vacuum used to control the placement of the batter. Those with enough artistic ability can make use of the fact that the earlier lines will cook longer and therefore be darker to include shading in their custom pancake. The PancakeBot does require a human helper to flip the pancake. The preordered units are expected to ship in the summer of 2015, and the retail price of the PancakeBot is $299.

Figure 2. The wireless, mobile home entertainment and monitoring KEECKER robot includes 360 degree surround sound and a projector that can rotate 90 degrees.

Jibo

Jibo, advertised as the world's first family robot, was launched through an Indiegogo campaign last summer. Cynthia Breazeal, the MIT professor who founded the robotics start-up company Jibo, is the creator of this futuristic, friendly home appliance. The company raised $2.3 million in the Indiegogo campaign, meeting all their stretch goals and receiving 4,800 preorders, which they hope to deliver by the end of 2015. In addition, the company announced in January that it raised $25.3 million in series A financing.

Jibo is about 11 inches tall and weighs in at six pounds, allowing users to easily move this robot around a home. Jibo includes two high-resolution cameras to track and recognize faces as well as capture photos and support video calls. Natural language processing lets users talk to Jibo as artificial intelligence algorithms will learn user preferences. Planned Jibo skills include serving as the family photographer by using natural cues to detect when someone is posing for a picture; telling interactive stories with graphics, sound effects, and physical movements; and acting as a personal assistant by reminding the family of important tasks and events.

During the Indiegogo campaign, backers were able to preorder the Home Edition for $499. The company plans to deliver the Jibo Developer Edition to their Indiegogo supporters and the beta release of the JiboAlive SDK in the fall of 2015. Visit www.jibo.com for more details.

KEECKER

KEECKER is a 16-inch tall, 26-pound home robot created by a Paris-based company of the same name. This egg-shaped robot is a home entertainment and monitoring system, with 360 degree surround sound and a 1,000 lumen, 720-pixel projector that can rotate 90 degrees to project on either a wall or the ceiling. A panoramic camera is positioned on top of the unit, providing a 360 degree view of the environment, and a 3D camera will allow KEECKER to map the house. This appliance contains numerous additional sensors including compass, accelerometer, gyroscope, microphone, ultrasound, infrared, light, air quality, temperature, and humidity. As opposed to Jibo, KEECKER is a mobile robot with two motorized wheels powered by two ultra-silent electric engines. KEECKER uses a lithium-ion battery, and the robot will detect when its battery is running low and will return to its contact-free induction charging base. Running on Android 5, KEECKER supports Wi-Fi a/c/b/g/n, Bluetooth 2.0, and BLE 4.0, and it can serve as a Wi-Fi access point. KEECKER can be preordered for $3,000, and units are expected to begin shipping in fall of 2015. For more details, visit www.keecker.com.

This article originally appeared in
IEEE Computer Graphics and Applications,
vol. 35, no. 3, 2015.

Lest We Forget

Vinton G. Cerf • *Google*

Recently, I've been focusing my attention on the preservation of digital information. This includes conventional text documents, images, videos, software, interactive programs, and webpages. I'm by no means the first person and certainly not the last to be interested in and concerned about our ability to preserve all kinds of digital content. One of the earliest efforts associated with the Internet and the World Wide Web was started and continues to be led by Brewster Kahle. His Internet Archive (https://archive.org/index.php) and associated Wayback Machine have been capturing Internet content for almost 20 years. It was started at a very early stage in the Internet's emergence to public view. He has enlisted the support of foundations, companies, and individuals and the Library of Alexandria in Egypt, among others.

As time goes on and as we become increasingly dependent on software to create, render, and interact with digital content, we'll also become increasingly dependent on our ability to preserve access to and to correctly interpret digitized information of all kinds. There's a primary challenge that is to preserve our ability to read the media in which we store digital information. I'm sure many readers have accumulated digital media for which they no longer have readers. Think of floppy disks, video tapes, 8-track audio tapes, and hard disk drives with interfaces that are no longer supported. Imagine what the US National Archives (www.archives.gov) must be faced with as it absorbs digital records on many media from the various government branches. What makes this such a challenge is that the digital files we seek to preserve could still be rendered unusable if we're unable to execute software that can correctly interpret the file formats and semantics of the information they contain.

The rate at which digital content is produced is increasing and it's not reasonable to assume that it can all be captured and preserved forever. That some information is important to save isn't debatable. Records that must be kept by law, such as records of real estate transactions, may be important for centuries. Many other records will be of uncertain value but experience has shown that some ancient records, preserved purely by happenstance, have proven to be enormously valuable, even if, at the time, they didn't seem so. This is true in some cases because the records cast light on earlier civilizations and societies and in other cases because they preserve knowledge. I think, for instance, of the so-called Dark Ages when much information was preserved only by monks' and Islamic scholars' dint of effort, among others. There are technologies that hold the promise of very long-term storage, but beg the question whether the devices needed to read the bits they contain will also be preserved or can be reconstructed. Even if we succeed in preserving access to the bits of digital information, we're also faced with preserving our ability to run the software that's needed for correct interpretation.

Open source software is often cited as a solution and it may well be for some cases. In other cases, we may need to preserve operating systems, application software, and detailed descriptions of the computer hardware needed to execute them. This is the approach taken by the Olive project (https://olivearchive.org/) at Carnegie-Mellon University that's directed by Mahadev Satyanarayanan. In a tour de force of technical effort, the Olive team has found a way to capture the details of hardware instruction sets and to capture snapshots of the loaded executables of the operating system and application software, allowing virtual machines to emulate

2376-113X/15/$31.00 © 2015 IEEE

older ones and to run software of the past to correctly render and interact with older content formats.

The implications of these ideas strongly suggest to me that we're in need of a "preservation regime" that affords privileges to parties working to preserve digital information. They might need the right to copy and execute software required for interpreting older files. They might need to be able to run virtual machines locally or in cloud-based systems as a service to users seeking to render or interact with older content. To be sure, there might also be ways to reformat older files into newer forms to match newer software or to maintain backward compatibility when new versions of software are created. But some companies will go out of business and others might choose not to support older software and formats. Even though not all content can or will be preserved, we should have the choice to do so if we wish, and that will require options among which to choose.

There might be other ways to preserve digital content by rendering it in alternative forms, such as printing on high rag-content paper or archival photographic stock. For those digital objects that have static representations, this may well be a useful alternative, but for content that requires software interpretation to use, we must consider methods like those developed by Satyanarayan and Kahle. Among the very big additional challenges is finding business models that allow long-term preservation of digital content over hundreds or thousands of years. That discussion will have to wait for another edition of *Backspace*!

Vinton G. Cerf is vice president and chief Internet evangelist at Google, and past president of ACM. He's widely known as one of the "fathers of the Internet." He's a fellow of IEEE and ACM. Contact him at vgcerf@gmail.com.

This article originally appeared in EEE Internet Computing, *vol. 19, no. 3, 2015.*

EIC's Message

Multimedia Search: From Relevance to Usefulness

Alan Hanjalic
Delft University of Technology

There has been an amazing amount of work done and progress achieved in the field of multimedia search over the past two decades. As nicely elaborated on by Lei Zhang and Yong Rui in their recent review of the advances in this field,[1] the developments so far have gone through three main stages: the text-based stage, the content-based stage, and the Web-based stage. In the text-based stage, multimedia search was essentially realized as a trivial extension of the classic, text-oriented information retrieval approach, where text was found in the documents accompanying multimedia items (images, video, and music). In the 1990s, researchers started to explore the possibilities of analyzing the actual (audio-visual) content of multimedia items to automatically infer semantic similarities and textual labels reflecting what is depicted in an image or a video frame and what is audible in a soundtrack. The main motivation for this stage was that manually adding texts to multimedia items was tedious and time consuming. Such texts were also typically one-sided and therefore not always informative enough to help locate the targeted multimedia item.

More recently, inspired by the rapid development of social media platforms, text has come back as an important information source for multimedia indexing and search; however, now it exists in the form of user-generated tags, comments (YouTube), and messages (tweets) accompanying multimedia items being uploaded on such platforms. Although still manually added, this "new text" has become scalable through social interaction. Furthermore, text inserted by various people covers different perspectives, increasing the richness of textual metadata. All this has helped text reenter the field so it can be increasingly exploited by the new wave of approaches marking the Web-based stage, where it is integrated with content-based analysis to make indexing and search more robust and reliable.

Different stages have been marked by the dominance of some key theoretical and algorithmic paradigms. For instance, the content-based stage started with an exploration of a broad range of audio-visual signal analysis methods,[2] but it later increasingly turned more toward well-established algorithmic frameworks, such as support vector machines (SVMs). Similarly, graph theory has provided the main analytic framework for the Web-based stage. The most recent algorithmic hype in the field is deep learning.

If we analyze these three stages and the related (dominating) technologies, we can conclude that their evolution has essentially been resource-driven. In other words, as we have moved from the text-based stage, via the content-based stage, to the Web-based stage, we have included more and more information resources in the development of methods for indexing and searching multimedia. Such technologies have then been investigated and enhanced to get the most out of these resources and improve the results of multimedia search in view of objective criteria, such as average precision (AP).

The key to understanding how a multimedia search system works lies in understanding the notion of relevance. Relevance has so far been explored and optimized mainly with respect to queries. The uncertainty in the channel connecting the query and the collection is typically large, and the need to overcome this uncertainty undoubtedly justifies the tremendous effort invested in the development of various relevance models over the past years,[3] involving more and more information resources and applying new generations of algorithms. An increasing number of results reported in recent literature indicate, however, that pursuing this resource-exploitation target of "technically" optimizing the relevance to a query may have already reached its limits. Specifically, it is unclear how much scientific breakthrough can still be achieved and how much impact all the

 2376-113X/15/$31.00

new proposed solutions will have in a practical technological context.

I believe it is time that we start looking more actively beyond relevance to the query and toward maximizing the usefulness of the search results for the user who inserted the query. This is not as straightforward as it may seem because, to do this, we also need to focus on the uncertainty between the user and the query—that is, to what extent does the query express the information need of the user?

Where does this uncertainty come from? Well, the user always has a reason for inserting a query, namely, a goal to do something with the obtained search results. In other words, there is always a particular search intent underlying the query insertion process. We can type in a query to locate content that might change our mood, for example, to find a funny picture or a video clip that will make us laugh. We may search for a multimedia document that will provide information about a topic of interest or a tutorial on how to do something (such as a video showing how to build a house). Finally, we may also search for a video taken by a head-mounted camera attached to a bungee jumper to help us immerse ourselves in the fantastic experience of a bungee jump, without actually having to jump ourselves.

All these reasons are difficult to express in a typically short query term, and they are also typically not (or insufficiently) encoded in the indexes of multimedia items. This is why query formulations for such examples are often done along the topical dimension to secure a match with a collection. The uncertainty related to the intent remains open, however, and with this also the risk of receiving inadequate search results. Interestingly, because of the topical bias of queries and rather mature solutions to filter out the items from a collection that do not match a topic, the received results are almost always "relevant" because they are related to the query's subject matter. Because of the uncertainty related to the user's intent, however, the returned relevant results are not necessarily useful in view of the user's true information need.

There are several interesting scientific challenges that the multimedia research community needs to face in handling the uncertainty inherent in search intent:

- Why do users search for multimedia items? A comprehensive set of search intent categories is needed.
- How do we assess whether a given multimedia item fits in a given intent category?
- How do we infer the proper intent category from the search session?
- How do we organize the search results to optimally fit the search intent?

The first explorative studies related to these challenges have already been reported in recent literature.[4,5] However, there is still a long road ahead of us to really understand users' needs in multimedia search and to develop systems that can optimally match these needs or, to formulate it more bluntly, to come up with multimedia search solutions that truly matter.[6] I sincerely hope to see more works along this new direction in the future. **MM**

References

1. L. Zhang and Y. Rui, "Image Search—From Thousands to Billions in 20 Years," *ACM Trans. Multimedia Computing, Comm. and Applications (TOMCCAP)*, vol. 9, no. 1, 2013, article no. 36.
2. A. Hanjalic, R.L. Lagendijk, and J. Biemond, "A New Key-Frame Allocation Method for Representing Stored Video Streams," *Proc. 1st Int'l Workshop on Image Databases and Multimedia Search*, 1996, pp. 64–67.
3. L. Yang, "Advanced Retrieval Models for Web Image Search," doctoral thesis, Delft Univ. of Technology, 2013.
4. A. Hanjalic, C. Kofler, and M.A. Larson, "Intent and its Discontents: The User at the Wheel of the Online Video Search Engine," *Proc. 20th ACM Int'l Conf. Multimedia*, 2012, pp. 1239–1248.
5. C. Kofler, M.A. Larson, and A. Hanjalic, "Intent-Aware Video Search Result Optimization," *IEEE Trans. Multimedia*, vol. 16, no. 5, 2014, pp. 1421–1433.
6. A. Hanjalic, "Multimedia Retrieval that Matters," *ACM Trans. Multimedia Computing, Comm. and Applications (TOMCCAP)*, vol. 9, no. 1s, 2013, article no. 44.

Alan Hanjalic is an associate editor in chief of *IEEE MultiMedia* and a professor of computer science and the head of the Multimedia Computing Group at the Delft University of Technology, The Netherlands. Contact him at a.hanjalic@tudelft.nl.

This article originally appeared in IEEE MultiMedia, *vol. 22, no. 1, 2015.*

The Need for New Business–Technology Relationships

Stephen J. Andriole, *Villanova School of Business*

A few years ago, I got into a car with the CEO of a large company. As I entered the car, he moved an object off the seat and placed it under his derriere. I asked him, "What did you just do?" He looked at me strangely and said "Nothing." I said, "Come on, when I got into the car you moved something under your derriere. What is it?" "If you must know," he said, "it's an iPad." Completely confused, I said, "Why are you hiding an iPad under your derriere?" He looked all around and said, "Because IT hasn't approved iPads."

Obviously, his company defined technology governance narrowly, and just as obviously, he ignored his company's governance policy, although he didn't want anyone to know, not even the consultants he hired to improve the business–technology relationship. In fact, this CEO was part of *shadow IT*—that is, IT spending that's accountable to no one, including the enterprise CIO.

Is this experience indicative of the old or new business–technology relationship? Let's look at trends in this relationship and where it's likely to be over the next five to 10 years.

New Technology Governance

For decades, the IT police wreaked havoc in companies where just thinking about a nonstandard hardware device or software application was a prosecutable offense. But how things have changed! The police are gone, and digital flowers are blooming everywhere: there's no need—except in very few companies—to hide devices from the technology standards police.

I described this new technology governance to the CFOs of 50 global corporations who visited the University of Pennsylvania's Wharton School last year to learn how to become better strategic partners with their business units. I told them that consumerization, democratization, and cloud delivery had redefined the rules around the acquisition, deployment, support, and measurement of technology, that there were now many participants in the governance process. Many of them weren't even employees of the companies buying (or renting) technology and were often vendors.

Not all of the CFOs liked what I had to say, but I came armed with survey and interview data that demonstrated that the worst way to manage IT is through centralized, organizational structures in which a control group dictates what's allowed and what's not.[1] In the old days, we called this *centralized governance*, but by the late 1990s and early 2000s, the world moved to *federated governance*, in which IT was jointly controlled by the enterprise and the business units.

Today, governance—no matter how much old CIOs and CFOs question it—is participatory, which means that the control of technology is shared among a variety of participants, including the vendors that provide cloud services and the independent developers of applications that run mostly on mobile devices.

New Device Deployment Models

At Villanova University, our undergraduates arrive on campus with iPhones and Mac Airs. They're very familiar with these devices and of course want to use them when they get to campus. A bad management practice would ban these devices because there's a history of, for example,

 Published by the IEEE Computer Society 2376-113X/15/$31.00

supporting Windows machines, and the IT team is therefore more familiar with Dells than Macs. (Fortunately, our CIO has a more flexible perspective on technology choices and allows Macs and Dells to live together in harmony.) The good news is that bring your own device (BYOD) deployment is gaining acceptance as more and more companies discover that their employees are more efficient when they're happy, and they're happiest when they get to play with their own digital toys. This is the new deployment model—and it's gaining traction.

New Skills and Competencies

How about the skills and competencies of the professionals responsible for cost-effectively sourcing, deploying, and measuring technology? For example, how many IT professionals really understand the nuances of cloud service-level agreements (CSLAs)? Or the differences between structured and unstructured data analytics? Or how the Internet of Things (IoT) will change supply chain optimization and big data injection, processing, storage, and analysis? New business technology managers are quickly developing skills, competencies, and best practices around CSLAs, the IoT, and related activities.

Many IT professionals also have precious little subject matter expertise about what their internal clients actually do. IT professionals who support pharmaceutical marketing need to know a great deal about what pharma marketers do and how technology can make them more productive. IT professionals who support insurance brokers, bankers, and police departments should know nearly all there is to know about how these industries work. Most IT teams are weak subject matter experts, pretty inadequate "SMEs." But this is changing. Business technologists are now dually informed about the business and the technology that can enable and extend business models and processes. Increasingly, CIOs are drawn from the business side of companies, not home-grown technologists whose previous jobs included running the company's Oracle database or help desk.

Old IT management also usually ignores horrible internal communications skills. Have you ever sat through a PowerPoint presentation that's essentially incoherent? We all have. Technologists are notoriously tone-deaf. Business units just want technology to work; they don't need to understand bits and bytes. They also need to know how their business models can be transformed with emerging technology. They don't need condescending lectures on what they should know about mobile communications, automated reasoning, or parallel processing. They need continuous, quick, simple, and impactful written, verbal, and video communications. That's IT for them!

Business units need to know how their business models can be transformed with emerging technology.

New Partnerships

The worst IT managers are the ones who still believe that technology and business are distinct business enablers, when today it's clear that there's no distinction between the two. In fact, they are one process with the same goals. CIOs and CTOs that see IT as somehow different from business models and processes have reached retirement age: it's impossible to separate or differentiate technology from business models and processes, or vice versa. Bad IT managers insist on perpetuating the us-versus-them mindset in continuous corporate combat, which is expensive and distracting.

New Organizational Structures

All of these trends will manifest themselves in new organizational structures. IT will bifurcate into operational and strategic parts. Operational IT will define and run the infrastructure—that is, email, office productivity tools (such as MS Office), backup, recovery, storage, security, and the like. Strategic IT will move to the business units that will select the software applications they need to profitably grow. The old enterprise CIO will become the chief infrastructure officer. CTOs will migrate to the business units as CDOs—chief digital officers. Eventually, operational technology will migrate to enterprise Audit, where many infrastructure activities will be deployed and certified. Both new CIOs and CTOs will rely almost exclusively on cloud providers to deliver operational and strategic technology.

IT's a Whole New World

Many enterprise CIOs are already stunned by the pace at which the changes I've described are occurring. In fact, according to industry analysts at Forrester and Gartner, these changes are gaining momentum. Years ago, the field was described as MIS (management information systems). Then it was described as IT. Now it's DT (digital technology) or DBT (digital business technology).

Companies are completely rethinking how they leverage technology for competitive advantage. What are the drivers of such dramatic change? Here are the primary ones and how they're changing IT:

- Demographics are killing IT.
- Consumerization is diluting IT and replacing technology innovation.
- Globalization is spreading IT.
- Cloud delivery is democratizing IT.
- Business processes are crushing IT.
- Vendors are confusing—and forcing—IT.
- Employees are ignoring IT.

Here are the outcomes:

- We will no longer access Web addresses; we will seek solutions.
- Expensive physical plants (such as offices) will be replaced by work-at-home, work-wherever/whenever-you-want, and BYOD models that indiscriminately cross personal and professional boundaries.
- Transportation infrastructure will simultaneously improve and shrink as digital communication improves and grows.
- Business processes will be unscheduled, personal, continuous, and mobile.
- Personal computers—and all single-purpose devices—will disappear.
- Web-enabled, intelligent wearables will be pervasive, integrate various technologies, and enable a variety of personal and professional activities.
- "Natural" interface technology will enable integration and efficiency.
- Everyone will be talking—and watching—all the time: social media will become pervasive.
- "Money" and traditional payment processes will disappear.

All of these drivers, trends, and outcomes have revealed themselves. More are on the way as technology evolves.

The new business–technology relationship is therefore likely to be based on integration and cooperation. All of the old animosities will disappear as the pace of technology change itself overwhelms old organizational traditions and structures. A full partnership is likely to emerge within the next five years, 10 at the latest. We all need to embrace the next phase of the business–technology relationship—especially since it's already here! IT

Reference

1. S.J. Andriole, "Who Owns IT?" *Comm. ACM*, vol. 58, no. 3, 2015, pp. 50–57.

Stephen J. Andriole *is the Thomas G. Labrecque Professor of Business Technology at the Villanova School of Business. His research interests include emerging technology adoption, technology management and governance, and unstructured data analytics. Andriole has published 30 books on technology trends and technology management, including* Ready Technology: Fast-Tracking New Business Technologies *(Auerbach Publications, 2014). More information on his career is available at www.andriole.com. Contact him at steve@andriole.com.*

This article originally appeared in IT Professional, *vol. 17, no. 4, 2015.*

Cognitive Assistance in the Workplace

Markus Funk and Albrecht Schmidt, *University of Stuttgart*

Some years ago, a customer could only choose already manufactured products from a predefined catalogue provided by a company. Today, customers can choose and configure nearly every detail of a product they want to buy. Each manufactured product thus becomes a unique piece rather than just another part in a large-scale production.

With online ordering, customized product orders can be transferred directly to the workplace where the product is being assembled, so the order is available for assembly the moment it's issued. The availability of such complex built-to-order systems also introduces a shift in how products are manufactured. A regular manufacturing workplace can hold spare parts for a variety of products, so a worker can assemble many different products at a single working place as the orders come in. Of course, this process of manufacturing is more complex and introduces more cognitive effort for the worker compared to a traditional built-to-stock approach. An increased cognitive load might also lead to more errors, and it might mean it takes longer to assemble a product. Moreover, there is still a need for supporting humans during assembly work, given the complexity, size, and cost of the specialized robots conducting assembly tasks.

The manual-assembly workplace is thus becoming a challenging environment. To help the workers better manage such environments, we, along with some of our colleagues and students, are building an assistive system that can be integrated directly into the workplace using in-situ projection for augmented reality (AR).

AN ASSISTIVE SYSTEM

We refer to our system as motionEAP, because it aims to increase *e*fficiency and *a*ssistance in *p*roduction processes based on *motion* detection and projection. It features a top-mounted projector that displays feedback regarding whether a worker selected the right parts and assembled those parts using appropriate tools (see Figure 1). The boxes, tools, and assembly positions are highlighted with a green light (see Figure 2). If the user picks a part from the wrong box or assembles the part incorrectly, the system flashes a red light.

In general, the assembly instructions are directly projected at the position where the action is required—for example, the light will flash on the box holding the correct part or on the location where the part should be placed. So the worker doesn't need to read the instructions from an external screen and transfer the perceived information into the real world.

Furthermore, a top-mounted depth camera provides continuous quality support. It can detect if a part was picked from the correct box and if a part was assembled correctly, letting the system immediately react to selection or assembly errors by displaying appropriate context-aware feedback. Using this technology, the system becomes "step aware," tracking the performed assembly steps, and it can advance the projected feedback when a part was assembled correctly, the correct tool was used, or the right part was picked from a box.

To ensure that all parts can be seen and highlighted by the top-mounted camera-projector pair, we designed a carrier for holding the assembled parts and designed the system to

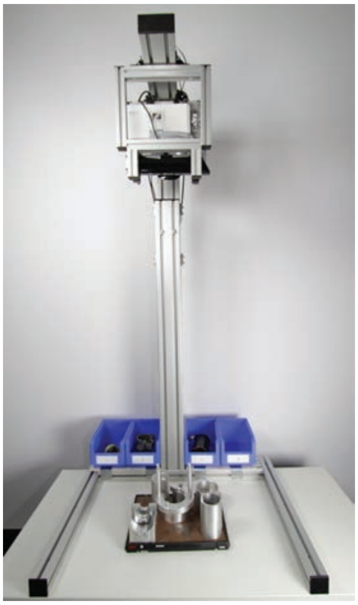


Figure 1. An assistive system that is integrated directly into the workplace using in-situ projection for augmented reality (AR).

SPOTLIGHT

Figure 2. A top-mounted projector displays feedback for picking right parts from the right box and correctly assembling those parts using the appropriate tools. The correct boxes, tools, and assembly positions are highlighted with a green light.

Figure 3. The system and its carrier while assembling a car's engine starter.

monitor the manufacturing from above. Figure 3 shows the system and its carrier while a worker assembles a car's engine starter. The system projects the position and orientation of the part to be assembled. The feedback only advances to the next step when depth data taken from the top-mounted depth camera matches the previously trained depth data for that particular step.

COGNITIVE SKILL TRANSFER

Traditionally, manufacturing skills have been transferred from experienced workers to inexperienced workers through repeated workflow demonstrations. Our prototype similarly learns the process by recording an experienced worker demonstrating the workflow. During the demonstration, our system continuously records which box to pick parts from, which tools to use, and where to assemble the parts. After the system has learned the workflow, it can independently project feedback according to the previously learned workflow, thereby teaching inexperienced workers. Using the previously recorded depth-data, it can check if the worker is performing the steps correctly, and because the recorded workflow can be saved to a file, it's transferrable between similar work places.

For example, when a manufacturing company outsources parts of its production process to another country, teaching new workflows can be cumbersome, and people often must physically travel to the remote company. Using our system, the workflow can be recorded in one place and transferred over the Internet.

In a study (currently under review for future publication), we compared our demonstration approach for creating interactive projected assembly instructions against videotaping the manufacturing processes and playing back the video. The results indicated that the workers using an in-situ projection approach learned the new assembly steps just as quickly as when learning from an assembly video. However, as the number of steps involved in the workflow increases, our in-situ approach helps people learn faster compared to learning from a video, and they make fewer errors.

INCREASING COGNITIVE ABILITIES

Our system can also support cognitively impaired workers as they perform their work tasks. To measure how cognitive abilities can be improved using our prototype, we conducted a study assembling Lego Duplo bricks (see Figure 4) with cognitively impaired workers. With the Duplo bricks, we could easily increase the complexity of the assembled product without changing the task.

The results show that using our in-situ projection approach, the time to assemble a part of a product and the errors made per assembled part stayed at a constant level, even when increasing the complexity of the working tasks. However, when using a pictorial approach, the time and errors increased with complexity. In another study with impaired workers, we coupled our assistive system with a machine used to produce clamps. The results indicated that the impaired workers benefited from projected in-situ feedback using a contour visualization.[1] More graphically complex visualizations, such as an assembly video or pictorial instructions, distracted the workers and lead to longer assembly times and more errors.

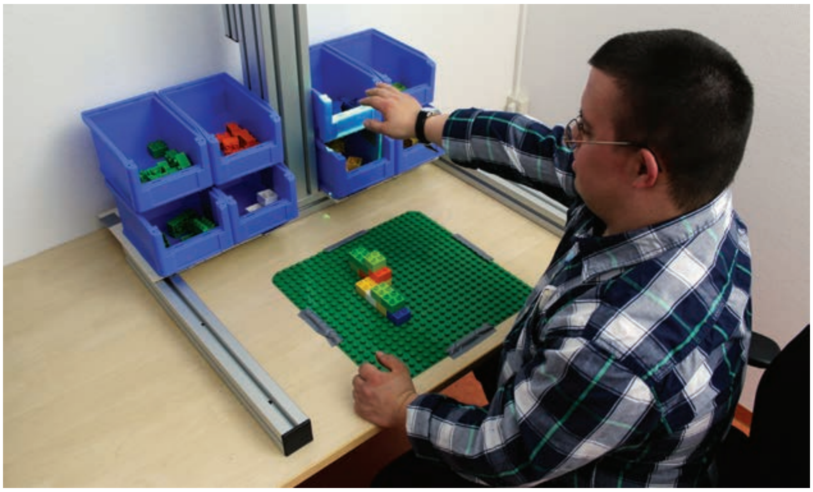


Figure 4. With the Duplo bricks, we could easily increase the complexity of the assembled product.

DOMAIN EXAMPLES

During our project, we deployed our prototype in the GWW (Gemeinnützige Werkstätten und Wohnstätten) sheltered work organization and in an industry setting at Audi AG. In the sheltered work organization, the system provided continuous cognitive support as people performed working tasks and was generally well received. One supervisor told us that one impaired worker hadn't completed a work task in five years, but our system motivated him to complete a series of steps. The participants also liked that the assistive system let workers pause the assembly at each working step and continue directly where they left off, without putting in extra cognitive effort to remember where they left off. According to Matthias Guth, the assistant general manager, "using the assistive system enables integrating workers with a strong impairment into the working process." He said that his employees are excited about and appreciate the system, and supervisors are seeing "a lower error rate when [employees are] conducting complex tasks."

In the industry setting, the system was mainly used to train new workers or to provide cognitive support when switching tasks. The trainees particularly liked the step awareness and the in-situ feedback. One participant stated that he "did not have to spend time transferring the instructions into the real world anymore." Another worker liked that "the system automatically switches from one working step to the other."

The results of already conducted studies show that using an assistive system like this at a working place to support cognitively impaired workers has a positive effect. In future research, we want to deploy our system in the sheltered work organization and conduct a study measuring the long-term effects of such assistive systems. Furthermore, we're interested in learning if there's a point in the workflow where an assistive system stops having a positive effect on the learning behavior in an industry scenario. Finally, we want to look into a mobile scenario, where the user is wearing the camera-projector pair. This would also let us monitor the assembly process from other viewpoints, not just from above the worker. Hopefully our assistive system using AR will greatly affect assembly workplaces in the future.

ACKNOWLEDGEMENTS

This work is funded by the German Ministry of Economics in the project MotionEAP, grant no. 01MT12021E.

REFERENCE

1. M. Funk et al., "Comparing Projected In-Situ Feedback at the Manual Assembly Workplace with Impaired Workers," *Proc. 8th Int'l Conf. Pervasive Technologies Related to Assistive Environments*, 2015.

Markus Funk is a PhD student in the human-computer interaction group at the University of Stuttgart. Contact him at markus.funk@vis.uni-stuttgart.de.

Albrecht Schmidt is a professor of human-computer interaction at the University of Stuttgart. Contact him at albrecht.schmidt@vis.uni-stuttgart.de.

This article originally appeared in IEEE Pervasive Computing, *vol. 14, no. 3, 2015.*

Enhancing Engineering Productivity

Douglass Post and Richard Kendall

The high level of technological innovation required for a strong national economy and defense is only achievable with a highly productive engineering workforce. In this country, the widely reported decline in the number of US students graduating in STEM disciplines strongly suggests that the US lead in innovation could be in jeopardy.[1] Changing demographics in the US engineering workforce exacerbate this situation: many of the most highly skilled engineers in the US defense community are rapidly approaching retirement age, and replacing their skills is a huge challenge.

This impact can be ameliorated by increasing engineering workforce productivity. Driven by US national defense needs[2,3] in the 1940s and 1950s, computers were initially developed to increase the productivity of scientists and mathematicians engaged in decryption and cryptography, the design and testing of nuclear weapons, the computation of artillery firing tables, the design and testing of aerospace systems—including the Apollo Program—and many other defense related programs. The adoption of PCs and workstations in the 1980s and 1990s accelerated office and engineering productivity via word processors, spreadsheets, business enterprise data processing, and computer-aided design systems. More recently, sophisticated engineering tools such as Matlab (www.mathworks.com) and Nastran, (www.mscsoftware.com/product/msc-nastran) as well as multi-physics "linked" tools such as ANSYS (www.ansys.com) and Comsol (www.comsol.com), have continued to increase engineering workforce productivity. However, the exponential growth in the computing power of laptops and small clusters that has sustained the growth of this productivity is slowing down and saturating.[4]

Fortunately, help is on the way. The confluence of the growing power of supercomputers and the development and deployment of multi-effect, science-based computational engineering software applications (tools) is beginning to give us the ability—for the first time in human history—to make accurate predictions of the performance of many complex, full-scale systems. For instance, the US Department of Defense High Performance Computing Modernization Program engineering application HPCMP CREATE-Kestrel can accurately predict the flight performance of a fixed-wing jet aircraft, including the aerodynamics, structural dynamics, propulsion, control, and other effects that determine the flight characteristics (http://aem.eng.ua.edu/files/2015/01/David-McDaniel-flyer.pdf). Similarly, Goodyear's tire design tool treats all the major effects that determine tire performance for modern vehicles.[5] Engineers now use validated tools like Kestrel and the Goodyear tire design tool to design and analyze fixed- and rotary-winged aircraft and modern complex tires, ships, complex antenna systems, ground vehicles, nuclear reactors, and many other complex systems. New tools are continually being developed for many other systems.

Going Virtual

All of these tools enable the construction and testing of virtual prototypes, thereby reducing the requirements for live tests and providing design decision data much earlier than previously achievable. Design engineers can develop optimized designs quickly as "virtual prototypes" and test them with multi-physics analysis tools early in the product development phase, even at the beginning of conceptual design and requirements definition. Design flaws, integration problems, and performance shortfalls can be detected and fixed before metal has been cut, thus avoiding time-consuming and costly rework.

 Published by the IEEE Computer Society

A classic example of the advantages of this paradigm for product development comes from Goodyear.[5] In 2003, Goodyear Tire began serious use of a set of multi-physics high-performance computing tools that could create virtual prototypes for complex, multi-tread, multi-layer tires and analyze and accurately predict their performance (tread wear, hydroplaning, low rolling resistance, and so on) prior to live testing. These tools enabled Goodyear to reduce its time to market by a factor of four, increase its innovation rate from 10 new products a year to more than 60, and reduce its testing costs by 60 percent (http://investor.goodyear.com/annuals.cfm).

Computational engineering design tools not only simulate a complex physical system's performance, their use can also simulate the product development process itself. By working on simulated design projects, engineers have the ability to gain extensive and wide-ranging work experience by participating in many, rather than just a few, design and development projects over the course of their career. Many large industries focus on a few major projects at any given time—for example, the design and product development of major airplanes, such as the Boeing 787, take nine or more years. For an engineering career that might span 35 years, an engineer can expect to be part of around four major projects. For military aircraft systems, the product development cycle time can take 25 years or more, which translates to just two projects over the course of a career. This undoubtedly overstates the severity of the problem, but the fact remains that having fewer major design projects reduces an engineer's ability to gain a wide range of work experience.

In this environment, it's difficult to get the kind of experience that engineers in the 1970s got when the product development cycle was closer to five years, enabling engineers to work on many more projects over their career. With computational tools to create and "test" virtual prototypes, engineers can develop, test, and optimize many different designs. This gives them feedback on how well their designs work and helps them develop the experience they need to advance their professional careers by giving them additional opportunities to grow their expertise and confidence.

Many large industries that produce complex systems are evolving into systems integrators. Much, if not most, of the technical product design and development is often outsourced to suppliers. The integrator needs engineers to oversee and manage the process of procuring and integrating components from suppliers into the system. These engineers must be knowledgeable and experienced and possess considerable technical judgment. But companies frequently, out of necessity, hire engineers who are early in their careers and haven't had time to acquire the skills, judgment, and confidence they need to be successful. The use of simulated design environments for training can accelerate the acquisition of such abilities, and it can also help with recruiting and retention. Engineers who start their first job expect to do "real" engineering that allows them to grow and mature their engineering and professional skills. If placed in a program management or procurement job that doesn't allow them to increase those skills, they often become dissatisfied and leave. The combination of technical work on a real project, combined with program management experience and participation in virtual design and development projects, can facilitate career growth, resulting in much greater job satisfaction and retention, as well as helping them improve their technical skills and their effectiveness as program managers.

Changing Demographics

The age distribution in many engineering organizations is bi-modal: most engineers are very senior and nearing retirement or are very young and inexperienced. These organizations need to cope with the imminent loss of the institutional knowledge and experience that resides in their senior engineering staff and rapidly transfer it from senior to junior staff. The use of a simulated design and product development process helps senior engineers mentor many more junior engineers in the early stage of their careers than they could in a conventional project. Catalogues of real and candidate designs can

This article originally appeared in Computing in Science & Engineering, *vol. 17, no. 4, 2015.*

be constructed, analyzed, and stored. Guided by a senior engineer, junior staff can use computational tools to analyze these designs and discuss the results with their mentor. Then they can develop their own designs, use the tools for analysis, and discuss them with their mentor. During our careers, we've seen one senior engineer train more than 10 engineers over the course of a year or two. The "graduates" of the simulated design and product development "school" eventually became a disproportionately large fraction of the leading engineers in that organization, and this training paradigm was gradually adopted across the board.

This paradigm also enables greater product innovation. In the traditional conceptual design process, an engineer usually constructs a few candidate designs and then iteratively refines those designs with a series of detailed analyses. This process is laborious and allows exploration of only a few design concepts. With computational physics-based conceptual design tools, thousands of options can be developed and assessed using low-fidelity analysis tools to weed out infeasible designs and identify the feasible ones. It's similar to Darwinian "natural selection." The low-fidelity tools are used to identify and eliminate the "less fit" designs—only the "fittest" designs survive. Variations of these "fittest" designs are developed and then subjected to the "natural selection" process. Once a final generation of "fittest" design options has been identified, more accurate high-fidelity tools can validate the choices made on the basis of low-fidelity analyses.

Of course, there are caveats. These tools are based on mathematical models of nature. They aren't nature, so they must be extensively verified and validated. Final predictions must be confirmed with live tests. These tools help to focus testing to make it more effective. This can reduce the amount of testing required, but doesn't replace testing. Quantitative knowledge of the uncertainties in the calculated results is needed to guide decisions. The codes must be used by experienced and knowledgeable subject matter experts because these tools aren't black boxes. It's all too easy for an inexperienced or careless user to get a faulty answer. However, in the hands of a skilled and knowledgeable engineer, these tools can greatly magnify productivity, just as the computers and decryption algorithms at Bletchley Park increased by many orders of magnitude[6] the speed, number, and fidelity of the decryptions that the British Ultra staff made of intercepted German military radio messages encrypted with Enigma machines. The importance of the workforce productivity enhancement provided by the tools and computers is described in the following quotation: "Sir Harry Hinsley, Bletchley Park veteran and official historian of British Intelligence in World War II, made a similar assessment about Ultra (the decryption work being done at Bletchley Park), saying that it shortened the war 'by not less than two years and probably by four years'; and that, in the absence of Ultra, it is uncertain how the war would have ended."[7] ■

References

1. N. Augustine et al., *Rising above the Gathering Storm, Revisited: Rapidly Approaching Category 5*, Nat'l Academies Press, 2010.
2. G. Dyson, *Turing's Cathedral*, Vintage, 2012.
3. L. Weiss, *America Inc., Innovation and Enterprise in the National Security State*, Cornell Univ. Press, 2014.
4. S. Fuller and L. Millett, *The Future of Computing Performance*, Nat'l Academies Press, 2011.
5. L.K. Miller, "Simulation-Based Engineering for Industrial Competitive Advantage," *Computing in Science & Eng.*, vol. 12, no. 3, 2010, pp. 14–21.
6. Andrew Hodges, *Alan Turning: The Enigma*, Princeton Univ. Press, 2014.
7. H. Hinsley, "The Influence of ULTRA in the Second World War," transcript of a lecture, Cambridge Univ., 19 Oct. 1993.

EDITOR: George Strawn, NITRD, gostrawn@gmail.com

Steve Jobs: IT Creative Genius

Jay Liebowitz, *Harrisburg University of Science and Technology*

Look around and you will see the effects that Steve Jobs and Apple have on our lives. Whether it's the iMac, iTunes, iPod, iPad, or iPhone, Jobs had a major role to play in these ubiquitous technology developments. In this light, Jobs is a modern-day IT hero and is well-deserving of this Mastermind column.

Flavor of the Person

The terms "visionary," "creative," and "innovator" are often associated with describing Jobs.[1] Others have also said that he was "demanding," a "perfectionist," and perhaps even "egotistical" at times.[2] Jobs was certainly an entrepreneur and a creative genius. Whether at NeXT, Pixar, or Apple, he had a vision for producing novel innovations with creative flair. He had an uncanny ability to take human factors and develop technology objects that would have mass appeal.

From the account of his biographers,[2] Jobs was a special kind of individual. Early in his schooling, he was a bit of a prankster, as he probably wasn't being challenged academically. In high school, he was introduced to Steve Wozniak ("Woz"). In 1976, the "two Steves" formed Apple Computer Company, and the fun commenced.[3]

As with most creative minds, Jobs saw things that others didn't. Take the iPod's rise, for example. Wherever you were—whether on the street, on a train, riding a bike, or jogging—people seemed to have these white things in their ears, listening to music. This also jumpstarted Apple's entry into the music industry. What foresight to envision this worldwide phenomenon!

All of Apple's products seem to have that artistic flair, yet a very comfortable feel to them. On a personal note, I have to confess that I was a Windows PC person for more than two decades. I was probably the last convert in my family to move from the PC to the Mac. However, once I did, I could never go back. The sleek look and intuitive,[4] easy-to-use features of the Mac made me a true believer.

It was unfortunate that Jobs's life was cut so short with pancreatic cancer. His contributions in the IT field are numerous, as I describe next.

IT Contributions (the iCon)

Jobs's legacy is omnipresent. Whether creating a tablet computer for mainly watching movies and playing games or using small-screen, multitouch technology for Apple products, Jobs is associated with developing products for mass consumption. Interestingly, in its early days and to cut expenses, Jobs eliminated Apple's research group, and was said to have focused more on development.[5] An interesting paradox exists here, as you'd think that Jobs and Apple would be strong advocates of research to develop their IT products. Certainly, and in later years, products such as the iPhone went beyond the typical chatting and emailing capacities of smartphones to Web surfing and "face-timing" capabilities. There are those who say that Apple is mainly product- rather than research-centric, and might have failed to extend its reach into the academic research market, as have Intel, Microsoft, HP, and others.[5] Whether this is true, Apple and Jobs certainly transformed lives around the world through their technological products.

This article originally appeared in IT Professional, *vol. 17, no. 3, 2015.*

A typical expression that's often heard is "follow your passion." In a Stanford University commencement speech, Jobs said the following:

> You've got to find what you love. ... [T]he only way to do great work is to love what you do. If you haven't found it yet, keep looking, and don't settle.[6]

Jobs had a perfect blend of interests in the liberal arts and technology to develop contributions to society. His Reed College liberal arts experience (although he never graduated), mixed with his excitement for the natural sciences, computers, and technology, thus created a special blend. This melting pot of disciplines and experiences led to such creative Apple products.

What better way to chronicle Jobs's contributions to the IT world than through the Smithsonian collections? As Megan Gambino points out in her *Smithsonian.com* article,[7] his contributions were many, and I highlight some as follows:

- The Apple II was the first personal computer to run a spreadsheet.
- Apple Macintosh introduced the GUI to the world.
- The iPhone, iPod Touch, and iPad reinvented the PDA for the Internet age.

Jobs had a knack for mixing his business savvy with design sensibility.[7] iTunes was also a great contribution in which people could play and organize digital music and video files on both their computers and portable devices.[8]

Failure and Lessons Learned

Through my knowledge management background, I have always been a strong believer in learning more from failures than successes. I actually started a journal titled *Failure and Lessons Learned in Information Technology Management*, which sadly, after three years, lived up to its name. However, I learned some important lessons. Similarly, great leaders like Jobs and Bill Gates are strong believers in experiential learning. In Gates' book, *The Road Ahead*,[9] he mentions that he would often look to hire senior executives for Microsoft who have gone through some business failures and recovered from them, to be better prepared in case similar tragedies occurred for Microsoft.

Jobs also had to endure some failures during his professional life. He was ousted from Apple in 1985. According to some,[2,10] Jobs lacked the emotional maturity and business discipline necessary to run a company such as Apple when he was only 30. During the hiatus between his departure and his return roughly 12 years later, Jobs learned some interesting lessons. According to Brent Schlender and Rick Tetzeli's new book,[11] Jobs learned about management skills from Pixar's founder Ed Catmull. He also learned to act less impulsively. However, despite these lessons, Schlender and Tetzeli point out through their interviews that Jobs was still perhaps too quick to judge people, which was a major shortcoming.

Most everyone will certainly agree that Jobs was a visionary and a great contributor to the IT field and society at large. In furthering this dream, Tim Cook, Apple's CEO, announced on 26 March 2015 that he planned to donate his own estimated US$785 million fortune to charity.[12] The Apple legacy continues as "giving back to the world." I believe that Steve Jobs would be proud.

References

1. J. Pepitone, "Steve Jobs: The Homepage Tributes," *CNN*, 6 Oct. 2011.
2. W. Isaacson, *Steve Jobs*, Simon & Schuster, 2011.
3. A. Brashares, *Steve Jobs: Thinking Different*, Twenty-First Century Books, 2001.
4. J. Liebowitz, ed., *Bursting the Big Data Bubble: The Case for Intuition-Based Decision Making*, Taylor & Francis, 2014.
5. S. Greengard, "What Did Steve Jobs Do for Computer Science?" *Comm. ACM*, Oct. 2011; http://cacm.acm.org/news/136161-what-did-steve-jobs-do-for-computer-science/fulltext.
6. C. Newport, *So Good They Can't Ignore You*, BusinessPlus, 2012.
7. M. Gambino, "The Many Contributions of Steve Jobs (1955–2011)," *Smithsonian.com*, 6 Oct. 2011; www.smithsonianmag.com/smithsonian-institution/the-many-contributions-of-steve-jobs-1955-2011-99643361/.
8. M. Wall, "Steve Jobs' Greatest Technology Contributions," *TechNews Daily*, 7 Oct. 2011; www.nbcnews.com/id/44817208/ns/technology_and_science-tech_and_gadgets/t/steve-jobs-greatest-technology-contributions/#.VRQhGGYegiQ.
9. B. Gates, *The Road Ahead*, Viking Press, 1995.
10. H. Blodget, "Let's Get One Thing Straight—Apple Had No Choice but to Oust Steve Jobs," *Business Insider*, 23 Sept. 2013; www.businessinsider.com/apple-had-no-choice-with-steve-jobs-2013-9.
11. B. Schlender and R. Tetzeli, *Becoming Steve Jobs: The Evolution of a Reckless Upstart into a Visionary Leader*, Random House, 2015.
12. Reuters, "Apple's Tim Cook to Donate All His Money, Magazine Says," *Fortune*, 26 Mar. 2015.

Jay Liebowitz *is the DiSanto Visiting Chair in Applied Business and Finance at Harrisburg University of Science and Technology. He previously was the Orkand Endowed Chair of Management and Technology at the University of Maryland University College, and a full professor at Johns Hopkins University. Contact him at jliebowitz@harrisburgu.edu.*

CAREER OPPORTUNITIES

MANAGE is seeking a Senior Software Engineer, Mobile & Infrastructure in San Francisco, CA to architect, implement, and maintain data pipelines and build predictive models to integrate with mobile devices. Ref Job ID: 9T5SMY & mail res. to Manage, Attn: HR, 660 Third Street, Suite 302, San Francisco, CA 94107.

PROGRAMMER ANALYST.: design, develop, test & implement application s/w utilizing knowledge of Java, J2EE, SQL, PL/SQL, JavaScript, HTML, Objective-C, iPhone/iPad app development, Servlets, JSP, EJB, Web Services, JavaBeans, Spring, Hibernate, Oracle, WebSphere, Weblogic, Windows, UNIX. Must be willing to travel & reloc. Reqs MS in comp sci, eng or rel. Mail resumes to Actium Consulting Inc. 308 Endless Court, Suite #1002, El Dorado Hills, CA. 95762.

NEWPORT CORP. seeks Product Engineer in Santa Clara, CA to enhance laser design to resolve yield issues and improve product reliability. RefJob Id: 9GFV49 & mail resume to Newport, Attn: S. Quintanilla, 3635 Peterson Way, Santa Clara, CA 95054.

ELECTRICAL ENGINEER - ROBOT SPECIALIST. Company/Institution: ATN Hoelzel LP, Responsible for the electrical design, inspection, testing and problem solving of automation equipment in automotive and solar industry. Provide technical support of electrical installations and ensure conformance to design specifications and compliance with safety and customer requirements. Minimum of 1 year work experience. Master's Degree in electrical engineering. Basic knowledge of Allen Bradley PLCs and Robotics. Ability to travel.

SAILPOINT TECHNOLOGIES INC. has an opening in Austin, TX for Software Test Engineer-as part of Agile QA team will guide & improve quality of cloud based SaaS product using enterprise bus. apps. Interested applicants mail resumes to Attn: A. Krupa (SP16), SailPoint, 11305 Four Points Drive, Bldg 2, Ste 100, Austin, TX 78726.

COMPUTER PROGRAMMER.: Create, modify, & test code, forms, & script & rewrite programs applying knowledge of computer capabilities, subject matter, & symbolic logic. Compile & write documentation of program development. Assist in software deployment & enforcement of software release standards for Java-based applications. Utilize Testing and Implementation of Oracle, Java, Windows XP and Quality center 10.0/9.0, UNIX. will work in unanticipated locations. Require 2 years experience. Send resume to Intriv Inc. 4625 Alexander Dr,# 125,Alpharetta, GA 30022.

VLOCITY is seeking a Senior Software Engineer in San Francisco, CA to develop Vlocity mobile applications for iPhone, iPad, Windows and Android devices. Ref Job ID: 9PBVYM & send res. To T. Dilley at hiring@vlocity.com

FRONT-END DEVELOPER position available w/ Satmetrix Systems, Inc. in Redwood City, CA. Turn designs into front-end code. M-F, 40 hrs/wk. Competitive sal & bnfts. To apply, please send resume to Satmetrix, Attn: Recruiting, 3 Twin Dolphin Drive, Suite 225, Redwood City, CA 94065. Reference job code: 9PGPAR. No calls.

OPTIMIZATION-BASED software company providing solutions for transportation scheduling & logistics problems has the following openings in Gainesville,

CLASSIFIED LINE AD SUBMISSION DETAILS: Rates are $425.00 per column inch ($640 minimum). Eight lines per column inch and average five typeset words per line. Send copy at least one month prior to publication date to: Debbie Sims, Classified Advertising, *Computer* Magazine, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; (714) 816-2138; fax (714) 821-4010. Email: dsims@computer.org.

In order to conform to the Age Discrimination in Employment Act and to discourage age discrimination, *Computer* may reject any advertisement containing any of these phrases or similar ones: "...recent college grads...," "...1–4 years maximum experience...," "...up to 5 years experience," or "...10 years maximum experience." *Computer* reserves the right to append to any advertisement without specific notice to the advertiser. Experience ranges are suggested minimum requirements, not maximums. *Computer* assumes that since advertisers have been notified of this policy in advance, they agree that any experience requirements, whether stated as ranges or otherwise, will be construed by the reader as minimum requirements only. *Computer* encourages employers to offer salaries that are competitive, but occasionally a salary may be offered that is significantly below currently acceptable levels. In such cases the reader may wish to inquire of the employer whether extenuating circumstances apply.

ADVERTISER INFORMATION • SEPTEMBER 2015

Advertising Personnel

Debbie Sims: Advertising Coordinator
Email: dsims@computer.org
Phone: +1 714 816 2138 | Fax: +1 714 821 4010

Chris Ruoff: Senior Sales Manager
Email: cruoff@computer.org
Phone: +1 714 816 2168 | Fax: +1 714 821 4010

Advertising Sales Representatives (display)

Central, Northwest, Far East:
Eric Kincaid
Email: e.kincaid@computer.org
Phone: +1 214 673 3742
Fax: +1 888 886 8599

Northeast, Midwest, Europe, Middle East:
David Schissler
Email: d.schissler@computer.org
Phone: +1 508 394 4026
Fax: +1 508 394 1707

Southwest, California:
Mike Hughes
Email: mikehughes@computer.org
Phone: +1 805 529 6790

Southeast:
Heather Buonadies
Email: h.buonadies@computer.org
Phone: +1 201 887 1703

Advertising Sales Representatives (Classifieds & Jobs Board)

Heather Buonadies
Email: h.buonadies@computer.org
Phone: +1 201 887 1703

C 100
Focus on
Your Job Search
IEEE Computer Society Jobs helps you easily find a new job in IT, software development, computer engineering, research, programming, architecture, cloud computing, consulting, databases, and many other computer-related areas.
New feature: Find jobs recommending or requiring the IEEE CS CSDA or CSDP certifications!
Visit www.computer.org/jobs to search technical job openings, plus internships, from employers worldwide.
http://www.computer.org/jobs
IEEE computer society | JOBS
The IEEE Computer Society is a partner in the AIP Career Network, a collection of online job sites for scientists, engineers, and computing professionals. Other partners include Physics Today, the American Association of Physicists in Medicine (AAPM), American Association of Physics Teachers (AAPT), American Physical Society (APS), AVS Science and Technology, and the Society of Physics Students (SPS) and Sigma Pi Sigma.