O'REILLY®

2014 Edition

Big Data Now

Gulfstream V
Savannah GA USA
Range 6,300 miles (10,139 km)
CAPACITY 14-19 passengers
MAX SPEED Mach 0.885
WINGSPAN 93 ft 4 in (28.5 m)
CREW 2 pilots, 0-2 attendants
WEIGHT 46,200 lb (21,000 kg)
FUEL SYSTEMS
WIND DIRECTION
WIND SPEED
GROUND SPEED

Current Perspectives from O'Reilly Media

Big Data Now

2014 Edition

O'Reilly Media, Inc.

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

Big Data Now: 2014 Edition
by O'Reilly Media, Inc.

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

Editor: Tim McGovern
Production Editor: Kristen Brown
Cover Designer: Ellie Volckhausen
Illustrator: Rebecca Demarest

January 2015: First Edition

Revision History for the First Edition
2015-01-09: First Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781491917367* for release details.

While the publisher and the author(s) have used good faith efforts to ensure that the information and instructions contained in this work are accurate, the publisher and the author(s) disclaim all responsibility for errors or omissions, including without limitation responsibility for damages resulting from the use of or reliance on this work. Use of the information and instructions contained in this work is at your own risk. If any code samples or other technology this work contains or describes is subject to open source licenses or the intellectual property rights of others, it is your responsibility to ensure that your use thereof complies with such licenses and/or rights.

978-1-491-91736-7

[LSI]

Table of Contents

Introduction: Big Data's Big Ideas

The big data space is maturing in dog years, seven years of maturity for each turn of the calendar. In the four years we have been producing our annual *Big Data Now*, the field has grown from infancy (or, if you prefer the canine imagery, an enthusiastic puppyhood) full of potential (but occasionally still making messes in the house), through adolescence, sometimes awkward as it figures out its place in the world, into young adulthood. Now in its late twenties, big data is now not just a productive member of society, it's a leader in some fields, a driver of innovation in others, and in still others it provides the analysis that makes it possible to leverage domain knowledge into scalable solutions.

Looking back at the evolution of our Strata events, and the data space in general, we marvel at the impressive data applications and tools now being employed by companies in many industries. Data is having an impact on business models and profitability. It's hard to find a non-trivial application that doesn't use data in a significant manner. Companies who use data and analytics to drive decision-making continue to outperform their peers.

Up until recently, access to big data tools and techniques required significant expertise. But tools have improved and communities have formed to share best practices. We're particularly excited about solutions that target new data sets and data types. In an era when the requisite data skill sets cut across traditional disciplines, companies have also started to emphasize the importance of processes, culture, and people.

As we look into the future, here are the main topics that guide our current thinking about the data landscape. We've organized this book around these themes:

Cognitive Augmentation

The combination of big data, algorithms, and efficient user interfaces can be seen in consumer applications such as Waze or Google Now. Our interest in this topic stems from the many tools that democratize analytics and, in the process, empower domain experts and business analysts. In particular, novel visual interfaces are opening up new data sources and data types.

Intelligence Matters

Bring up the topic of algorithms and a discussion on recent developments in artificial intelligence (AI) is sure to follow. AI is the subject of an ongoing series of posts on O'Reilly Radar. The "unreasonable effectiveness of data" notwithstanding, algorithms remain an important area of innovation. We're excited about the broadening adoption of algorithms like deep learning, and topics like feature engineering, gradient boosting, and active learning. As intelligent systems become common, security and privacy become critical. We're interested in efforts to make machine learning secure in adversarial environments.

The Convergence of Cheap Sensors, Fast Networks, and Distributed Computing

The Internet of Things (IoT) will require systems that can process and unlock massive amounts of event data. These systems will draw from analytic platforms developed for monitoring IT operations. Beyond data management, we're following recent developments in streaming analytics and the analysis of large numbers of time series.

Data (Science) Pipelines

Analytic projects involve a series of steps that often require different tools. There are a growing number of companies and open source projects that integrate a variety of analytic tools into coherent user interfaces and packages. Many of these integrated tools enable replication, collaboration, and deployment. This remains an active area, as specialized tools rush to broaden their coverage of analytic pipelines.

The Evolving, Maturing Marketplace of Big Data Components

Many popular components in the big data ecosystem are open source. As such, many companies build their data infrastructure and products by assembling components like Spark, Kafka, Cassandra, and ElasticSearch, among others. Contrast that to a few years ago when many of these components weren't ready (or didn't exist) and companies built similar technologies from scratch. But companies are interested in applications and analytic platforms, not individual components. To that end, demand is high for data engineers and architects who are skilled in maintaining robust data flows, data storage, and assembling these components.

Design and Social Science

To be clear, data analysts have always drawn from social science (e.g., surveys, psychometrics) and design. We are, however, noticing that many more data scientists are expanding their collaborations with product designers and social scientists.

Building a Data Culture

"Data-driven" organizations excel at using data to improve decision-making. It all starts with instrumentation. "If you can't measure it, you can't fix it," says DJ Patil, VP of product at RelateIQ. In addition, developments in distributed computing over the past decade have given rise to a group of (mostly technology) companies that excel in building data products. In many instances, data products evolve in stages (starting with a "minimum viable product") and are built by cross-functional teams that embrace alternative analysis techniques.

The Perils of Big Data

Every few months, there seems to be an article criticizing the hype surrounding big data. Dig deeper and you find that many of the criticisms point to poor analysis and highlight issues known to experienced data analysts. Our perspective is that issues such as privacy and the cultural impact of *models* are much more significant.

Cognitive Augmentation

We address the theme of cognitive augmentation first because this is where the rubber hits the road: we build machines to make our lives better, to bring us capacities that we don't otherwise have—or that only some of us would. This chapter opens with Beau Cronin's thoughtful essay on predictive APIs, things that deliver the right functionality and content at the right time, for the right person. The API is the interface that tackles the challenge that Alistair Croll defined as "Designing for Interruption." Ben Lorica then discusses graph analysis, an increasingly prevalent way for humans to gather information from data. Graph analysis is one of the many building blocks of cognitive augmentation; the way that tools interact with each other—and with us—is a rapidly developing field with huge potential.

Challenges Facing Predictive APIs

Solutions to a number of problems must be found to unlock PAPI value

by Beau Cronin

In November, the first International Conference on Predictive APIs and Apps will take place in Barcelona, just ahead of Strata Barcelona. This event will bring together those who are building intelligent web services (sometimes called Machine Learning as a Service) with those who would like to use these services to build predictive apps, which, as defined by Forrester, deliver "the right functionality and content at the right time, for the right person, by continuously learning about them and predicting what they'll need."

This is a very exciting area. Machine learning of various sorts is revolutionizing many areas of business, and predictive services like the ones at the center of predictive APIs (PAPIs) have the potential to bring these capabilities to an even wider range of applications. I cofounded one of the first companies in this space (acquired by Salesforce in 2012), and I remain optimistic about the future of these efforts. But the field as a whole faces a number of challenges, for which the answers are neither easy nor obvious, that must be addressed before this value can be unlocked.

In the remainder of this post, I'll enumerate what I see as the most pressing issues. I hope that the speakers and attendees at PAPIs will keep these in mind as they map out the road ahead.

Data Gravity

It's widely recognized now that for truly large data sets, it makes a lot more sense to move compute to the data rather than the other way around—which conflicts with the basic architecture of cloud-based analytics services such as predictive APIs. It's worth noting, though, that after transformation and cleaning, many machine learning data sets are actually quite small—not much larger than a hefty spreadsheet. This is certainly an issue for the truly big data needed to train, say, deep learning models.

Workflow

The data gravity problem is just the most basic example of a number of issues that arise from the development process for data science and data products. The Strata conferences right now are flooded with proposals from data science leaders who stress the iterative and collaborative nature of this work. And it's now widely appreciated that the preparatory (data preparation, cleaning, transformation) and communication (visualization, presentation, storytelling) phases usually consume far more time and energy than model building itself. The most valuable toolsets will directly support (or at least not disrupt) the whole process, with machine learning and model building closely integrated into the overall flow. So, it's not enough for a predictive API to have solid client libraries and/or a slick web interface: instead, these services will need to become upstanding, fully assimilated citizens of the existing data science stacks.

Crossing the Development/Production Divide

Executing a data science *project* is one thing; delivering a robust and scalable data *product* entails a whole new set of requirements. In a nutshell, project-based work thrives on flexible data munging, tight iteration loops, and lightweight visualization; productization emphasizes reliability, efficient resource utilization, logging and monitoring, and solid integration with other pieces of distributed architecture. A predictive API that supports one of these endeavors won't necessarily shine in the other setting. These limitations might be fine if expectations are set correctly; it's fine for a tool to support, say, exploratory work, with the understanding that production use will require re-implementation and hardening. But I do think the reality does conflict with some of the marketing in the space.

Users and Skill Sets

Sometimes it can be hard to tell at whom, exactly, a predictive service is aimed. Sophisticated and competent *data scientists*—those familiar with the ins and outs of statistical modeling and machine learning methods—are typically drawn to high-quality open source libraries, like scikit-learn, which deliver a potent combination of control and ease of use. For these folks, predictive APIs are likely to be viewed as opaque (if the methods aren't transparent and flexible) or of questionable value (if the same results could be achieved using a free alternative). *Data analysts*, skilled in data transformation and manipulation but often with limited coding ability, might be better served by a more integrated "workbench" (such as those provided by legacy vendors like SAS and SPSS). In this case, the emphasis is on the overall experience rather than the API. Finally, *application developers* probably just want to add predictive capabilities to their products, and need a service that doesn't force them to become de facto (and probably subpar) data scientists along the way.

These different needs are conflicting, and clear thinking is needed to design products for the different personas. But even that's not enough: the real challenge arises from the fact that developing a single data product or predictive app will often require all three kinds of effort. Even a service that perfectly addresses one set of needs is therefore at risk of being marginalized.

Horizontal versus Vertical

In a sense, all of these challenges come down to the question of *value*. What aspects of the total value chain does a predictive service address? Does it support ideation, experimentation and exploration, core development, production deployment, or the final user experience? Many of the developers of predictive services that I've spoken with gravitate naturally toward the *horizontal* aspect of their services. No surprise there: as computer scientists, they are at home with abstraction, and they are intellectually drawn to—even entranced by—the underlying similarities between predictive problems in fields as diverse as finance, health care, marketing, and e-commerce. But this perspective is misleading if the goal is to deliver a solution that carries more value than free libraries and frameworks. Seemingly trivial distinctions in language, as well as more fundamental issues such as appetite for risk, loom ever larger.

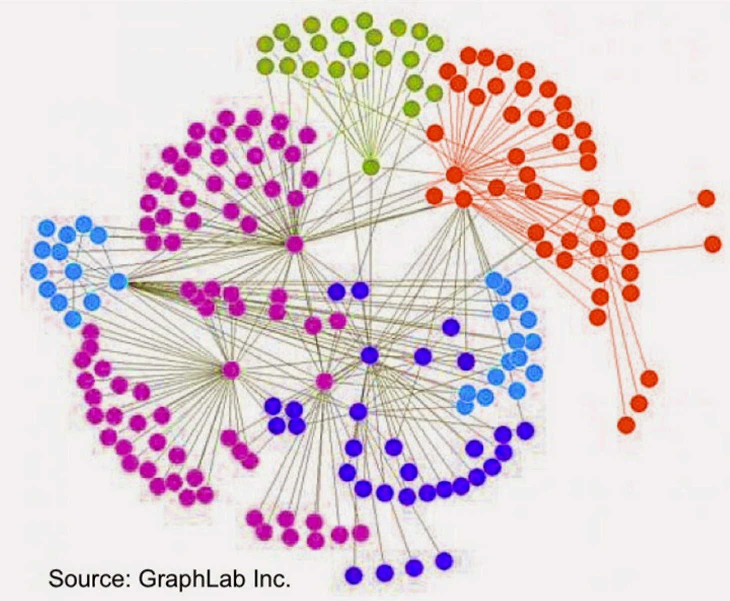
As a result, predictive API providers will face increasing pressure to specialize in one or a few verticals. At this point, elegant and general APIs become not only irrelevant, but a potential liability, as industry- and domain-specific feature engineering increases in importance and it becomes crucial to present results in the right parlance. Sadly, these activities are not thin adapters that can be slapped on at the end, but instead are ravenous time beasts that largely determine the perceived value of a predictive API. No single customer cares about the generality and wide applicability of a platform; each is looking for the best solution to the problem as he conceives it.

As I said, I am hopeful that these issues can be addressed—if they are confronted squarely and honestly. The world is badly in need of more accessible predictive capabilities, but I think we need to enlarge the problem before we can truly solve it.

There Are Many Use Cases for Graph Databases and Analytics

Business users are becoming more comfortable with graph analytics

by Ben Lorica



The rise of sensors and connected devices will lead to applications that draw from network/graph data management and analytics. As the number of devices surpasses the number of people—Cisco estimates 50 *billion* connected devices by 2020—one can imagine applications that depend on data stored in graphs with many more nodes and edges than the ones currently maintained by social media companies.

This means that researchers and companies will need to produce real-time tools and techniques that scale to much larger graphs (measured in terms of nodes and edges). I previously listed tools for tapping into graph data, and I continue to track improvements in accessibility, scalability, and performance. For example, at the justconcluded Spark Summit, it was apparent that GraphX remains a high-priority project within the Spark[1] ecosystem.

Another reason to be optimistic is that tools for graph data are getting tested in many different settings. It's true that social media applications remain natural users of graph databases and analytics. But there are a growing number of applications outside the "social" realm. In his recent Strata Santa Clara talk and book, Neo Technology's founder and CEO Emil Eifrem listed other uses cases for graph databases and analytics:

- Network impact analysis (including root cause analysis in data centers)
- Route finding (going from point A to point B)
- Recommendations
- Logistics
- Authorization and access control
- Fraud detection
- Investment management and finance (including securities and debt)

The widening number of applications means that business users are becoming more comfortable with graph analytics. In some domains network science dashboards are beginning to appear. More recently, analytic tools like GraphLab Create make it easier to unlock and build applications with graph[2] data. Various applications that build upon graph search/traversal are becoming common, and users are beginning to be comfortable with notions like "centrality" and "community structure".

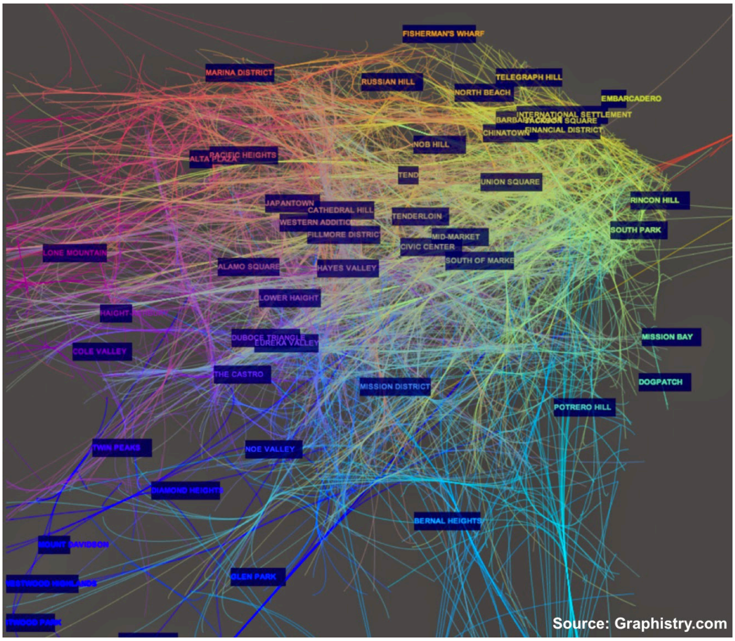
A quick way to immerse yourself in the graph analysis space is to attend the third GraphLab conference in San Francisco—a showcase of the best tools[3] for graph data management, visualization, and analytics, as well as interesting use cases. For instance, MusicGraph will be on hand to give an overview of their massive graph database from the music industry, Ravel Law will demonstrate how they leverage

[1] Full disclosure: I am an advisor to Databricks—a startup commercializing Apache Spark.

[2] As I noted in a previous post, GraphLab has been extended to handle general machine learning problems (not just graphs).

[3] Exhibitors at the GraphLab conference will include creators of several major graph databases, visualization tools, and Python tools for data scientists.

graph tools and analytics to improve search for the legal profession, and Lumiata is assembling a database to help improve medical science using evidence-based tools powered by graph analytics.



Source: Graphistry.com

Figure 1-1. Interactive analyzer of Uber trips across San Francisco's micro-communities

Network Science Dashboards

Network graphs can be used as primary visual objects with conventional charts used to supply detailed views

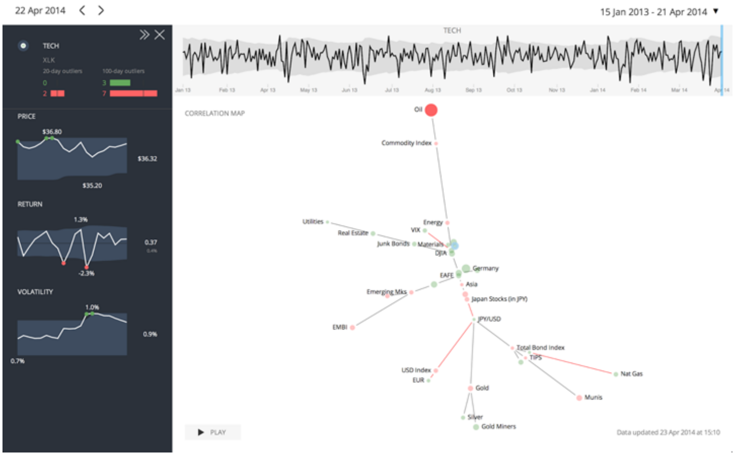
by Ben Lorica

With Network Science well on its way to being an established academic discipline, we're beginning to see tools that leverage it.[4] Applications that draw heavily from this discipline make heavy use of visual representations and come with interfaces aimed at business users. For business analysts used to consuming bar and line charts, network visualizations take some getting used. But with enough practice, and for the right set of problems, they are an effective visualization model.

In many domains, networks graphs can be the primary visual objects with conventional charts used to supply detailed views. I recently got a preview of some dashboards built using Financial Network Analytics (FNA). In the example below, the primary visualization represents correlations among assets across different asset classes[5] (the accompanying charts are used to provide detailed information for individual nodes):

[4] This post is based on a recent conversation with Kimmo Soramäki, founder of Financial Network Analytics.

[5] Kimmo is an experienced researcher and policy-maker who has consulted and worked for several central banks. Thus FNA's first applications are aimed at financial services.



Using the network graph as the center piece of a dashboard works well in this instance. And with FNA's tools already being used by a variety of organizations and companies in the financial sector, I think "Network Science dashboards" will become more commonplace in financial services.

Network Science dashboards only work to the extent that network graphs are effective (networks graphs tend get harder to navigate and interpret when the number of nodes and edges get large[6]). One workaround is to aggregate nodes and visualize communities rather than individual objects. New ideas may also come to the rescue: the rise of networks and graphs is leading to better techniques for visualizing large networks.

This fits one of the themes we're seeing in Strata: *cognitive augmentation*. The right combination of data/algorithm(s)/interface allows analysts to make smarter decisions much more efficiently. While much of the focus has been on data and algorithms, it's good to see more emphasis paid to effective interfaces and visualizations.

[6] Traditional visual representations of large networks are pejoratively referred to as "hairballs."

Intelligence Matters

Artificial intelligence has been "just around the corner" for decades. But it's more accurate to say that our ideas of what we can expect from AI have been sharpening and diversifying since the invention of the computer. Beau Cronin starts off this chapter with consideration of AI's 'dueling definitions'—and then resolves the "duel" by considering both artificial and human intelligence as part of a system of knowledge; both parts are vital and new capacities for both human and machine intelligence are coming.

Pete Warden then takes us through deep learning—one form of machine intelligence whose performance has been astounding over the past few years, blasting away expectations particularly in the field of image recognition. Mike Loukides then brings us back to the big picture: what makes human intelligence is not power, but the desire for betterment.

AI's Dueling Definitions

Why my understanding of AI is different from yours

by Beau Cronin

Let me start with a secret: I feel self-conscious when I use the terms "AI" and "artificial intelligence." Sometimes, I'm downright embarrassed by them.

Before I get into why, though, answer this question: what pops into *your* head when you hear the phrase artificial intelligence?

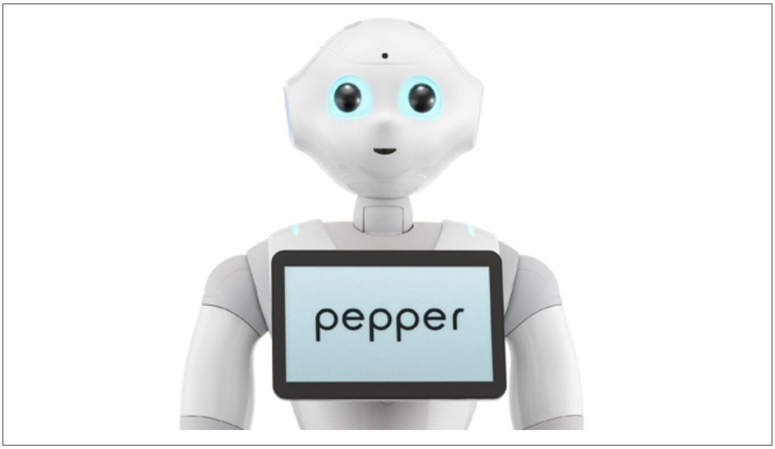


Figure 2-1. SoftBank's Pepper, a humanoid robot that takes its surroundings into consideration.

For the layperson, AI might still conjure HAL's unblinking red eye, and all the misfortune that ensued when he became so tragically confused. Others jump to the replicants of *Blade Runner* or more recent movie robots. Those who have been around the field for some time, though, might instead remember the "old days" of AI—whether with nostalgia or a shudder—when intelligence was thought to primarily involve logical reasoning, and truly intelligent machines seemed just a summer's work away. And for those steeped in today's big-data-obsessed tech industry, "AI" can seem like nothing more than a high-falutin' synonym for the machine-learning and predictive-analytics algorithms that are already hard at work optimizing and personalizing the ads we see and the offers we get—it's the term that gets trotted out when we want to put a high sheen on things.

Like the Internet of Things, Web 2.0, and big data, AI is discussed and debated in many different contexts by people with all sorts of motives and backgrounds: academics, business types, journalists, and technologists. As with these other nebulous technologies, it's no wonder the meaning of AI can be hard to pin down; everyone sees what they want to see. But AI also has serious historical baggage, layers of meaning and connotation that have accreted over generations of university and industrial research, media hype, fictional accounts, and funding cycles. It's turned into a real problem: without a lot of context, it's impossible to know what someone is talking about when they talk about AI.

Let's look at one example. In his 2004 book *On Intelligence*, Jeff Hawkins confidently and categorically states that AI failed decades ago. Meanwhile, the data scientist John Foreman can casually discuss the "AI models" being deployed every day by data scientists, and Marc Andreessen can claim that enterprise software products have already achieved AI. It's such an overloaded term that all of these viewpoints are valid; they're just starting from different definitions.

Which gets back to the embarrassment factor: I know what I mean when I talk about AI, at least I think I do, but I'm also painfully aware of all these other interpretations and associations the term evokes. And I've learned over the years that the picture in my head is almost always radically different from that of the person I'm talking to. That is, what drives all this confusion is the fact that different people rely on different primal archetypes of AI.

Let's explore these archetypes, in the hope that making them explicit might provide the foundation for a more productive set of conversations in the future.

AI as interlocutor

This is the concept behind both HAL and Siri: a computer we can talk to in plain language, and that answers back in our own lingo. Along with Apple's personal assistant, systems like Cortana and Watson represent steps toward this ideal: they aim to meet us on our own ground, providing answers as good as—or better than—those we could get from human experts. Many of the most prominent AI research and product efforts today fall under this model, probably because it's such a good fit for the search- and recommendation-centric business models of today's Internet giants. This is also the version of AI enshrined in Alan Turing's famous test for machine intelligence, though it's worth noting that direct assaults on that test have succeeded only by gaming the metric.

AI as android

Another prominent notion of AI views disembodied voices, however sophisticated their conversational repertoire, as inadequate: witness the androids from movies like *Blade Runner*, *I Robot*, *Alien*, *The Terminator*, and many others. We routinely transfer our expectations from these fictional examples to real-world efforts like Boston Dynamics' (now Google's) Atlas, or SoftBank's newly announced Pepper. For many practitioners and enthusiasts, AI simply *must* be mechanically embodied to fulfill the true ambitions of the field. While there is a body of theory to motivate this insistence, the attachment to mechanical form seems more visceral, based on a collective gut feeling that intelligences must move and act in the world to be worthy of our attention. It's worth noting that, just as recent Turing test results have highlighted the degree to which people are willing to ascribe intelligence to conversation partners, we also place unrealistic expectations on machines with human form.

AI as reasoner and problem-solver

While humanoid robots and disembodied voices have long captured the public's imagination, whether empathic or psychopathic, early AI pioneers were drawn to more refined and high-minded tasks—playing chess, solving logical proofs, and planning complex tasks. In a much-remarked collective error, they mistook the tasks that were hardest for smart humans to perform (those that seemed by introspection to require the most intellectual effort) for those that would be hardest for machines to replicate. As it turned out, computers excel at these kinds of highly abstract, well-defined jobs. But they struggle at the things we take for granted—things that children and many animals perform expertly, such as smoothly navigating the physical world. The systems and methods developed for games like chess are completely useless for real-world tasks in more varied environments.Taken to its logical conclusion, though, this is the scariest version of AI for those who warn about the dangers of artificial superintelligence. This stems from a definition of intelligence that is "an agent's ability to achieve goals in a wide range of environments." What if an AI was as good at general problem-solving as Deep Blue is at chess? Wouldn't that AI be likely to turn those abilities to its own improvement?

AI as big-data learner

This is the ascendant archetype, with massive amounts of data being inhaled and crunched by Internet companies (and governments). Just as an earlier age equated machine intelligence with the ability to hold a passable conversation or play chess, many current practitioners see AI in the prediction, optimization, and recommendation systems that place ads, suggest prod-

> ucts, and generally do their best to cater to our every need and commercial intent. This version of AI has done much to propel the field back into respectability after so many cycles of hype and relative failure—partly due to the profitability of machine learning on big data. But I don't think the predominant machine-learning paradigms of classification, regression, clustering, and dimensionality reduction contain sufficient richness to express the problems that a sophisticated intelligence must solve. This hasn't stopped AI from being used as a marketing label—despite the lingering stigma, this label is reclaiming its marketing mojo.

This list is not exhaustive. Other conceptualizations of AI include the superintelligence that might emerge—through mechanisms never made clear—from a sufficiently complex network like the Internet, or the result of whole-brain emulation (i.e., mind uploading).

Each archetype is embedded in a deep mesh of associations, assumptions, and historical and fictional narratives that work together to suggest the technologies most likely to succeed, the potential applications and risks, the timeline for development, and the "personality" of the resulting intelligence. I'd go so far as to say that it's impossible to talk and reason about AI without reference to *some* underlying characterization. Unfortunately, even sophisticated folks who should know better are prone to switching mid-conversation from one version of AI to another, resulting in arguments that descend into contradiction or nonsense. This is one reason that much AI discussion is so muddled—we quite literally don't know what we're talking about.

For example, some of the confusion about deep learning stems from it being placed in multiple buckets: the technology has proven itself successful as a big-data learner, but this achievement leads many to assume that the same techniques can form the basis for a more complete interlocutor, or the basis of intelligent robotic behavior. This confusion is spurred by the Google mystique, including Larry Page's stated drive for conversational search.

It's also important to note that there are possible intelligences that fit *none* of the most widely held stereotypes: that are not linguistically sophisticated; that do not possess a traditional robot embodiment;

that are not primarily goal driven; and that do not sort, learn, and optimize via traditional big data.

Which of these archetypes do I find most compelling? To be honest, I think they all fall short in one way or another. In my next post, I'll put forth a new conception: AI as model-building. While you might find yourself disagreeing with what I have to say, I think we'll at least benefit from having this debate explicitly, rather than talking past each other.

In Search of a Model for Modeling Intelligence

True artificial intelligence will require rich models that incorporate real-world phenomena

by Beau Cronin

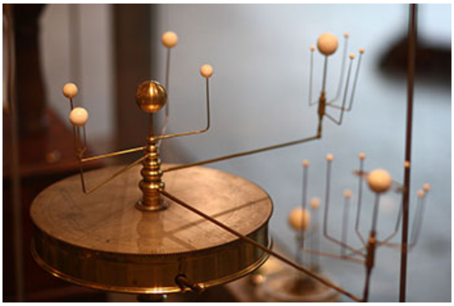


Figure 2-2. An orrery, a runnable model of the solar system that allows us to make predictions. Photo: Wikimedia Commons.

In my last post, we saw that AI means a lot of things to a lot of people. These dueling definitions each have a deep history—OK fine, baggage—that has massed and layered over time. While they're all legitimate, they share a common weakness: each one can apply perfectly well to a system that is not particularly intelligent. As just one example, the chatbot that was recently touted as having passed the Turing test is certainly an interlocutor (of sorts), but it was widely criticized as not containing any significant intelligence.

Let's ask a different question instead: What criteria must *any* system meet in order to achieve intelligence—whether an animal, a smart robot, a big-data cruncher, or something else entirely?

To answer this question, I want to explore a hypothesis that I've heard attributed to the cognitive scientist Josh Tenenbaum (who was a member of my thesis committee). He has not, to my knowledge, unpacked this deceptively simple idea in detail (though see his excellent and accessible paper How to Grow a Mind: Statistics, Structure, and Abstraction), and he would doubtless describe it quite differently from my attempt here. Any foolishness which follows is therefore most certainly my own, and I beg forgiveness in advance.

I'll phrase it this way:

Intelligence, whether natural or synthetic, derives from a model of the world in which the system operates. Greater intelligence arises from richer, more powerful, "runnable" models that are capable of more accurate and contingent predictions about the environment.

What do I mean by a model? After all, people who work with data are always talking about the "predictive models" that are generated by today's machine learning and data science techniques. While these models do technically meet my definition, it turns out that the methods in wide use capture very little of what is knowable and important about the world. We can do much better, though, and the key prediction of this hypothesis is that systems will gain intelligence proportionate to how well the models on which they rely incorporate additional aspects of the environment: physics, the behaviors of other intelligent agents, the rewards that are likely to follow from various actions, and so on. And the most successful systems will be those whose models are "runnable," able to reason about and simulate the consequences of actions without actually taking them.

Let's look at a few examples.

- Single-celled organisms leverage a simple behavior called chemotaxis to swim toward food and away from toxins; they do this by detecting the relevant chemical concentration gradients in their liquid environment. The organism is thus acting on a simple model of the world—one that, while devastatingly simple, usually serves it well.
- Mammalian brains have a region known as the hippocampus that contains cells that fire when the animal is in a particular place, as well as cells that fire at regular intervals on a hexagonal

grid. While we don't yet understand all of the details, these cells form part of a system that models the physical world, doubtless to aid in important tasks like finding food and avoiding danger —not so different from the bacteria.

- While humans also have a hippocampus, which probably performs some of these same functions, we also have overgrown neocortexes that model many other aspects of our world, including, crucially, our social environment: we need to be able to predict how others will act in response to various situations.

The scientists who study these and many other examples have solidly established that naturally occurring intelligences rely on internal models. The question, then, is whether artificial intelligences must rely on the same principles. In other words, what exactly did we mean when we said that intelligence "derives from" internal models? Just how strong is the causal link between a system having a rich world model and its ability to possess and display intelligence? Is it an absolute dependency, meaning that a sophisticated model is a *necessary condition* for intelligence? Are good models merely *very helpful* in achieving intelligence, and therefore likely to be present in the intelligences that we build or grow? Or is a model-based approach but *one path among many* in achieving intelligence? I have my hunches—I lean toward the stronger formulations—but I think these need to be considered open questions at this point.

The next thing to note about this conception of intelligence is that, bucking a long-running trend in AI and related fields, it is not a behavioralist measure. Rather than evaluating a system based on its actions alone, we are affirmedly piercing the veil in order to make claims about what is happening on the inside. This is at odds with the most famous machine intelligence assessment, the Turing test; it also contrasts with another commonly-referenced measure of general intelligence, "an agent's ability to achieve goals in a wide range of environments".

Of course, the reason for a naturally-evolving organism to spend significant resources on a nervous system that can build and maintain a sophisticated world model is to generate actions that promote reproductive success—big brains are energy hogs, and they need to pay rent. So, it's not that behavior doesn't matter, but rather that the strictly behavioral lens might be counterproductive if we want to learn *how* to build generally intelligent systems. A focus on the input-output characteristics of a system might suffice when its goals are relatively narrow, such as medical diagnoses, question answering, and image classification (though each of these domains could benefit from more sophisticated models). But this black-box approach is necessarily descriptive, rather than normative: it describes a desired endpoint, without suggesting how this result should be achieved. This devotion to surface traits leads us to adopt methods that do not not scale to harder problems.

Finally, what does this notion of intelligence say about the current state of the art in machine intelligence as well as likely avenues for further progress? I'm planning to explore this more in future posts, but note for now that today's most popular and successful machine learning and predictive analytics methods—deep neural networks, random forests, logistic regression, Bayesian classifiers—all produce models that are remarkably impoverished in their ability to represent real-world phenomena.

In response to these shortcomings, there are several active research programs attempting to bring richer models to bear, including but not limited to probabilistic programming and representation learning. By now, you won't be surprised that I think such approaches represent our best hope at building intelligent systems that can truly be said to understand the world they live in.

Untapped Opportunities in AI

Some of AI's viable approaches lie outside the organizational boundaries of Google and other large Internet companies

by Beau Cronin

Figure 2-3. Photo: GOC Bengeo to Woodhall Park 159: Woodhall Park boundary wall by Peter aka anemoneprojectors, on Flickr

Here's a simple recipe for solving crazy-hard problems with machine intelligence. First, collect huge amounts of training data—probably more than anyone thought sensible or even possible a decade ago. Second, massage and preprocess that data so the key relationships it contains are easily accessible (the jargon here is "feature engineering"). Finally, feed the result into ludicrously high-performance, parallelized implementations of pretty standard machine-learning methods like logistic regression, deep neural networks, and k-means clustering (don't worry if those names don't mean anything to you—the point is that they're widely available in high-quality open source packages).

Google pioneered this formula, applying it to ad placement, machine translation, spam filtering, YouTube recommendations, and even the self-driving car—creating billions of dollars of value in the process. The surprising thing is that Google isn't made of magic. Instead, mirroring Bruce Scheneier's surprised conclusion about the NSA in the wake of the Snowden revelations, "its tools are no different from what we have in our world; it's just better funded."

Google's success is astonishing not only in scale and diversity, but also the degree to which it exploded the accumulated conventional wisdom of the artificial intelligence and machine learning fields. Smart people with carefully tended arguments and closely held theories about how to build AI were proved wrong (not the first time this happened). So was born the *unreasonable* aspect of data's effectiveness: that is, the discovery that simple models fed with very large datasets really crushed the sophisticated theoretical approaches that were all the rage before the era of big data.

In many cases, Google has succeeded by reducing problems that were previously assumed to require *strong AI*—that is, reasoning and problem-solving abilities generally associated with human intelligence—into *narrow AI*, solvable by matching new inputs against vast repositories of previously encountered examples. This alchemy rests critically on step one of the recipe above: namely, acquisition of data at scales previously rejected as absurd, if such collection was even considered before centralized cloud services were born.

Now the company's motto makes a bit more sense: "Google's mission is to organize the world's information and make it universally accessible and useful." Yes, to machines. The company's ultimate success relies on transferring the rules and possibilities of the online world to our physical surroundings, and its approach to machine learning and AI reflects this underlying drive.

But is it the only viable approach? With Google (and other tech giants) buying robotics and AI companies at a manic clip—systematically moving into areas where better machine learning will provide a compelling advantage and employing "less than 50% but certainly more than 5%" of ML experts—it's tempting to declare game over. But, with the caveat that we know little about the company's many unannounced projects (and keeping in mind that I have approximately zero insider info), we can still make some good guesses about areas where the company, and others that have adopted its model, are unlikely to dominate.

I think this comes down to situations that have one or more of the following properties:

1. *The data is inherently small* (for the relevant definition of small) and further collection is illegal, prohibitively expensive or even impossible. Note that this is a high bar: sometimes a data collection scheme that seems out of reach is merely waiting for the appropriate level of effort and investment, such as driving down every street on earth with a specially equipped car.
2. *The data really cannot be interpreted without a sophisticated model*. This is tricky to judge, of course: the unreasonable effectiveness of data is exactly that it exposes just how superfluous models are in the face of simple statistics computed over large datasets.
3. *The data cannot be pooled* across users or customers, whether for legal, political, contractual, or other reasons. This results in many "small data" problems, rather than one "big data" problem.

My friend and colleague Eric Jonas points out that genomic data is a good example of properties one and two. While it might seem strange to call gene sequencing data "small," keep in mind there are "only" a few billion human genomes on earth, each comprising a few billion letters. This means the vast majority of possible genomes—including many perfectly good ones—will never be observed; on the other hand, those that do exist contain enough letters that plenty of the patterns we find will turn out to be misleading: the product of chance, rather than a meaningfully predictive signal (a problem called over-fitting). The disappointing results of genome-wide association studies, the relatively straightforward statistical analyses of gene sequences that represented the first efforts to identify genetic predictors of disease, reinforce the need for approaches that incorporate more knowledge about how the genetic code is read and processed by cellular machinery to produce life.

Another favorite example of mine is perception and autonomous navigation in *unknown* environments. Remember that Google's cars would be completely lost anywhere without a pre-existing high-resolution map. While this might scale up to handle everyday driving in many parts of the developed world, many autonomous vehicle or robot applications will require the system to recognize and understand its environment from scratch, and adapt to novel challenges in real time. What about autonomous vehicles exploring new territory for the first time (think about an independent Mars rover, at one extreme), or that face rapidly-shifting or even adversarial situations in which a static map, however detailed, simply can't capture the essential aspects of the situation? The bottom line is that there are many environments that can't be measured or instrumented sufficiently to be rendered legible to Google-style machines.

Other candidates include the interpretation and prediction of company performance from financial and other public data (properties 1 and 2); understanding manufacturing performance and other business processes directly from sensor data, and suggesting improvements thereon (2 and 3); and mapping and optimizing the real information and decision-making flows within organizations, an area that's seen far more promise than delivery (1, 2, and 3).

This is a long way from coherent advice, but it's in areas like these where I see the opportunities. It's not that the large Internet companies *can't* go after these applications; it's that these kinds of problems fit poorly with their ingrained assumptions, modes of organization, existing skill sets, and internal consensus about the right way to go about things. Maybe that's not much daylight, but it's all you're going to get.

What is Deep Learning, and Why Should You Care?

by Pete Warden

When I first ran across the results in the Kaggle image-recognition competitions, I didn't believe them. I've spent years working with machine vision, and the reported accuracy on tricky tasks like distinguishing dogs from cats was beyond anything I'd seen, or imagined I'd see anytime soon. To understand more, I reached out to one of the competitors, Daniel Nouri, and he demonstrated how he used the Decaf open-source project to do so well. Even better, he showed me how he was quickly able to apply it to a whole bunch of other image-recognition problems we had at Jetpac, and produce much better results than my conventional methods.

I’ve never encountered such a big improvement from a technique that was largely unheard of just a couple of years before, so I became obsessed with understanding more. To be able to use it commercially across hundreds of millions of photos, I built my own specialized library to efficiently run prediction on clusters of low-end machines and embedded devices, and I also spent months learning the dark arts of training neural networks. Now I’m keen to share some of what I’ve found, so if you’re curious about what on earth deep learning is, and how it might help you, I’ll be covering the basics in a series of blog posts here on Radar, and in a short upcoming ebook.

So, What is Deep Learning?

It’s a term that covers a particular approach to building and training neural networks. Neural networks have been around since the 1950s, and like nuclear fusion, they’ve been an incredibly promising laboratory idea whose practical deployment has been beset by constant delays. I’ll go into the details of how neural networks work a bit later, but for now you can think of them as decision-making black boxes. They take an array of numbers (that can represent pixels, audio waveforms, or words), run a series of functions on that array, and output one or more numbers as outputs. The outputs are usually a prediction of some properties you’re trying to guess from the input, for example whether or not an image is a picture of a cat.

The functions that are run inside the black box are controlled by the memory of the neural network, arrays of numbers known as weights that define how the inputs are combined and recombined to produce the results. Dealing with real-world problems like cat-detection requires very complex functions, which mean these arrays are very large, containing around 60 million numbers in the case of one of the recent computer vision networks. The biggest obstacle to using neural networks has been figuring out how to set all these massive arrays to values that will do a good job transforming the input signals into output predictions.

Training

One of the theoretical properties of neural networks that has kept researchers working on them is that they should be teachable. It’s pretty simple to show on a small scale how you can supply a series of example inputs and expected outputs, and go through a mechanical process to take the weights from initial random values to progressively better numbers that produce more accurate predictions (I'll give a practical demonstration of that later). The problem has always been how to do the same thing on much more complex problems like speech recognition or computer vision with far larger numbers of weights.

That was the real breakthrough in the 2012 Imagenet paper sparking the current renaissance in neural networks. Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton brought together a whole bunch of different ways of accelerating the learning process, including convolutional networks, clever use of GPUs, and some novel mathematical tricks like ReLU and dropout, and showed that in a few weeks they could train a very complex network to a level that outperformed conventional approaches to computer vision.

This isn't an aberration, similar approaches have been used very successfully in natural language processing and speech recognition. This is the heart of deep learning—the new techniques that have been discovered that allow us to build and train neural networks to handle previously unsolved problems.

How is it Different from Other Approaches?

With most machine learning, the hard part is identifying the features in the raw input data, for example SIFT or SURF in images. Deep learning removes that manual step, instead relying on the training process to discover the most useful patterns across the input examples. You still have to make choices about the internal layout of the networks before you start training, but the automatic feature discovery makes life a lot easier. In other ways, too, neural networks are more general than most other machine-learning techniques. I've successfully used the original Imagenet network to recognize classes of objects it was never trained on, and even do other image tasks like scene-type analysis. The same underlying techniques for architecting and training networks are useful across all kinds of natural data, from audio to seismic sensors or natural language. No other approach is nearly as flexible.

Why Should You Dig In Deeper?

The bottom line is that deep learning works really well, and if you ever deal with messy data from the real world, it's going to be an essential element in your toolbox over the next few years. Until recently, it's been an obscure and daunting area to learn about, but its success has brought a lot of great resources and projects that make it easier than ever to get started. I'm looking forward to taking you through some of those, delving deeper into the inner workings of the networks, and generally have some fun exploring what we can all do with this new technology!

Artificial Intelligence: Summoning the Demon

We need to understand our own intelligence is competition for our artificial, not-quite intelligences

by Mike Loukides

In October, Elon Musk likened artificial intelligence (AI) to "summoning the demon". As I'm sure you know, there are many stories in which someone summons a demon. As Musk said, they rarely turn out well.

There's no question that Musk is an astute student of technology. But his reaction is misplaced. There are certainly reasons for concern, but they're not Musk's.

The problem with AI right now is that its achievements are greatly over-hyped. That's not to say those achievements aren't real, but they don't mean what people think they mean. Researchers in deep learning are happy if they can recognize human faces with 80% accuracy. (I'm skeptical about claims that deep learning systems can reach 97.5% accuracy; I suspect that the problem has been constrained some way that makes it much easier. For example, asking "is there a face in this picture?" or "where is the face in this picture?" is much different from asking "what is in this picture?") That's a hard problem, a really hard problem. But humans recognize faces with nearly 100% accuracy. For a deep learning system, that's an almost inconceivable goal. And 100% accuracy is orders of magnitude harder than 80% accuracy, or even 97.5%.

What kinds of applications can you build from technologies that are only accurate 80% of the time, or even 97.5% of the time? Quite a few. You might build an application that created dynamic travel guides from online photos. Or you might build an application that measures how long diners stay in a restaurant, how long it takes them to be served, whether they're smiling, and other statistics. You might build an application that tries to identify who appears in your photos, as Facebook has. In all of these cases, an occasional error (or even a frequent error) isn't a big deal. But you wouldn't build, say, a face-recognition-based car alarm that was wrong 20% of the time—or even 2% of the time.

Similarly, much has been made of Google's self-driving cars. That's a huge technological achievement. But Google has always made it very clear that their cars rely on the accuracy of their highly detailed street view. As Peter Norvig has said, it's a hard problem to pick a traffic light out of a scene and determine if it is red, yellow, or green. It is trivially easy to recognize the color of a traffic light that you already know is there. But keeping Google's street view up to date isn't simple. While the roads change infrequently, towns frequently add stop signs and traffic lights. Dealing with these changes to the map is extremely difficult, and only one of many challenges that remain to be solved: we know how to interpret traffic cones, we know how to think about cars or humans behaving erratically, we know what to do when the lane markings are covered by snow. That ability to think like a human when something unexpected happens makes a self-driving car a "moonshot" project. Humans certainly don't perform perfectly when the unexpected happens, but we're surprisingly good at it.

So, AI systems can do, with difficulty and partial accuracy, some of what humans do all the time without even thinking about it. I'd guess that we're 20 to 50 years away from anything that's more than a crude approximation to human intelligence. It's not just that we need bigger and faster computers, which will be here sooner than we think. We don't understand how human intelligence works at a fundamental level. (Though I wouldn't assume that understanding the brain is a prerequisite for artificial intelligence.) That's not a problem or a criticism, it's just a statement of how difficult the problems are. And let's not misunderstand the importance of what we've accomplished: this level of intelligence is already extremely useful. Computers don't get tired, don't get distracted, and don't panic. (Well, not often.) They're great for assisting or augmenting human intelligence, precisely because as an assistant, 100% accuracy isn't required. We've had cars with computer-assisted parking for more than a decade, and they've gotten quite good. Larry Page has talked about wanting Google search to be like the Star Trek computer, which can understand context and anticipate what the humans wants. The humans remain firmly in control, though, whether we're talking to the Star Trek computer or Google Now.

I'm not without concerns about the application of AI. First, I'm concerned about what happens when humans start relying on AI systems that really aren't all that intelligent. AI researchers, in my experience, are fully aware of the limitations of their systems. But their customers aren't. I've written about what happens when HR departments trust computer systems to screen resumes: you get some crude pattern matching that ends up rejecting many good candidates. Cathy O'Neil has written on several occasions about machine learning's potential for dressing up prejudice as "science."

The problem isn't machine learning itself, but users who uncritically expect a machine to provide an oracular "answer," and faulty models that are hidden from public view. In a not-yet published paper, DJ Patil and Hilary Mason suggest that you search Google for GPS and cliff; you might be surprised at the number of people who drive their cars off cliffs because the GPS told them to. I'm not surprised; a friend of mine owns a company that makes propellers for high-performance boats, and he's told me similar stories about replacing the propellers for clients who run their boats into islands.

David Ferrucci and the other IBMers who built Watson understand that Watson's potential in medical diagnosis isn't to have the last word, or to replace a human doctor. It's to be part of the conversation, offering diagnostic possibilities that the doctor hasn't considered, and the reasons one might accept (or reject) those diagnoses. That's a healthy and potentially important step forward in medical treatment, but do the doctors using an automated service to help make diagnoses understand that? Does our profit-crazed health system understand that? When will your health insurance policy say "you can only consult a doctor after the AI has failed"? Or "Doctors are a thing of the past, and if the AI is wrong 10% of the time, that's acceptable; after all, your doctor wasn't right all the time, anyway"? The problem isn't the tool; it's the application of the tool. More specifically, the problem is forgetting that an assistive technology is assistive, and assuming that it can be a complete stand-in for a human.

Second, I'm concerned about what happens if consumer-facing researchers get discouraged and leave the field. Although that's not likely now, it wouldn't be the first time that AI was abandoned after a wave of hype. If Google, Facebook, and IBM give up on their "moonshot" AI projects, what will be left? I have a thesis (which may eventually become a Radar post) that a technology's future has a lot to do with its origins. Nuclear reactors were developed to build bombs, and as a consequence, promising technologies like Thorium reactors were abandoned. If you can't make a bomb from it, what good is it?

If I'm right, what are the implications for AI? I'm thrilled that Google and Facebook are experimenting with deep learning, that Google is building autonomous vehicles, and that IBM is experimenting with Watson. I'm thrilled because I have no doubt that similar work is going on in other labs, in other places, that we know nothing about. I don't want the future of AI to be shortchanged because researchers hidden in government labs choose not to investigate ideas that don't have military potential. And we do need a discussion about the role of AI in our lives: what are its limits, what applications are OK, what are unnecessarily intrusive, and what are just creepy. That conversation will never happen when the research takes place behind locked doors.

At the end of a long, glowing report about the state of AI, Kevin Kelly makes the point that every advance in AI, every time computers make some other achievement (playing chess, playing Jeopardy, inventing new recipes, maybe next year playing Go), we redefine the meaning of our own human intelligence. That sounds funny; I'm certainly suspicious when the rules of the game are changed every time it appears to be "won," but who really wants to define human intelligence in terms of chess-playing ability? That definition leaves out most of what's important in humanness.

Perhaps we need to understand our own intelligence is competition for our artificial, not-quite intelligences. And perhaps we will, as Kelly suggests, realize that maybe we don't really want "artificial intelligence." After all, human intelligence includes the ability to be wrong, or to be evasive, as Turing himself recognized. We want "artificial smartness": something to assist and extend our cognitive capabilities, rather than replace them.

That brings us back to "summoning the demon," and the one story that's an exception to the rule. In Goethe's Faust, Faust is admitted to heaven: not because he was a good person, but because he never ceased striving, never became complacent, never stopped trying to figure out what it means to be human. At the start, Faust mocks Mephistopheles, saying "What can a poor devil give me? When has your kind ever understood a Human Spirit in its highest striving?" (lines 1176-7, my translation). When he makes the deal, it isn't the typical "give me everything I want, and you can take my soul"; it's "When I lie on my bed satisfied, let it be over…when I say to the Moment 'Stay! You are so beautiful,' you can haul me off in chains" (1691-1702). At the end of this massive play, Faust is almost satisfied; he's building an earthly paradise for those who strive for freedom every day, and dies saying "In anticipation of that blessedness, I now enjoy that highest Moment" (11585-6), even quoting the terms of his deal.

So, who's won the bet? The demon or the summoner? Mephistopheles certainly thinks he has, but the angels differ, and take Faust's soul to heaven, saying "Whoever spends himself striving, him we can save" (11936-7). Faust may be enjoying the moment, but it's still in anticipation of a paradise that he hasn't built. Mephistopheles fails at luring Faust into complacency; rather, he is the driving force behind his striving, a comic figure who never understands that by trying to drag Faust to hell, he was pushing him toward humanity. If AI, even in its underdeveloped state, can serve this function for us, calling up that demon will be well worth it.

The Convergence of Cheap Sensors, Fast Networks, and Distributed Computing

One of the great drivers in the development of big data technologies was the explosion of data inflows to central processing points, and the increasing demand for outputs from those processing points—Google's servers, or Twitter's, or any number of other server-based technologies. This shows no signs of letting up; on the contrary, we are creating new tools (and toys) that create data every day, with accelerometers, cameras, GPS units, and more.

Ben Lorica kicks off this chapter with a review of tools for dealing with this ever-rising flood of data. Then Max Shron gives a data scientist's perspective on the world of hardware data: what unique opportunities and constraints does data produced by things that are 99% machine, 1% computer bring? How to create value from flows of data is the next topic, covered by Mike Barlow, and then Andy Oram covers a specific use case: smarter buildings. Finally, in a brief coda, Alistair Croll looks ahead to see even more distribution of computing, with more independence of devices and computation at the edges of the network.

Expanding Options for Mining Streaming Data

New tools make it easier for companies to process and mine streaming data sources

Stream processing was in the minds of a few people that I ran into over the past week. A combination of new systems, deployment tools, and enhancements to existing frameworks, are behind the recent chatter. Through a combination of simpler deployment tools, programming interfaces, and libraries, recently released tools make it easier for companies to process and mine streaming data sources.

Of the distributed stream processing systems that are part of the Hadoop ecosystem,[1] Storm is by far the most widely used (more on Storm below). I've written about Samza, a new framework from the team that developed *Kafka* (an extremely popular messaging system). Many companies who use Spark express interest in using Spark Streaming (many have already done so). Spark Streaming is distributed, fault-tolerant, *stateful*, and boosts programmer productivity (the same code used for batch processing can, with minor tweaks, be used for realtime computations). But it targets applications that are in the "second-scale latencies." Both Spark Streaming and Samza have their share of adherents and I expect that they'll both start gaining deployments in 2014.

Netflix[2] recently released Suro, a data pipeline service for large volumes of event data. Within Netflix, it is used to dispatch events for both batch (Hadoop) and realtime (Druid) computations:

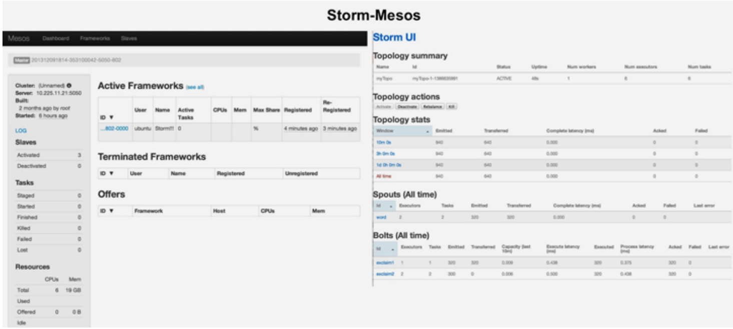
Leveraging and Deploying Storm

YARN, Storm and Spark were key to letting Yahoo! move from batch to near realtime analytics. To that end, Yahoo! and Hortonworks built tools that let Storm applications leverage Hadoop clusters. Mesosphere released a similar project for Mesos (Storm-Mesos) early this week. This release makes it easier to run Storm on Mesos clusters: in particular, previous Storm-Mesos integrations do not have the "built-in configuration distribution" feature and require a lot more steps deploy. (Along with several useful tutorials on how to

[1] New batch of commercial options: In-memory grids (e.g., Terracotta, ScaleOut software, GridGain) also have interesting stream processing technologies.

[2] Some people think Netflix is building other streaming processing components.

run different tools on top of Mesos, Mesosphere also recently released Elastic Mesos.)


Storm-Mesos

201312091814-353100042-5050-802

Cluster: (Unnamed)
Server: 10.225.11.21:5050
Built: 2 months ago by root
Started: 6 hours ago

Slaves

Activated	3
Deactivated	0

Tasks

Staged	0
Started	0
Finished	0
Killed	0
Failed	0
Lost	0

Resources

	CPUs	Mem
Total	6	19 GB
Used		
Offered	0	0 B
Idle		

Active Frameworks

ID ▼	User	Name	Active Tasks	CPUs	Mem	Max Share	Registered	Re-Registered
...802-0000	ubuntu	Storm!!!	0			%	4 minutes ago	3 minutes ago

Terminated Frameworks

ID ▼	User	Name	Registered	Unregistered

Offers

ID ▼	Framework	Host	CPUs	Mem

Storm UI

Topology summary

Name	Id	Status	Uptime	Num workers	Num executors	Num tasks
myTopo	myTopo-1-1386635991	ACTIVE	48s	1	6	6

Topology actions

Topology stats

Window	Emitted	Transferred	Complete latency (ms)	Acked	Failed
10m 0s	940	640	0.000	0	0
3h 0m 0s	940	640	0.000	0	0
1d 0h 0m 0s	940	640	0.000	0	0
All time	940	640	0.000	0	0

Spouts (All time)

Id	Executors	Tasks	Emitted	Transferred	Complete latency (ms)	Acked	Failed	Last error
word	2	2	320	320	0.000	0	0	

Bolts (All time)

Id	Executors	Tasks	Emitted	Transferred	Capacity (last 10m)	Execute latency (ms)	Executed	Process latency (ms)	Acked	Failed	Last error
exclaim1	1	1	320	320	0.009	0.438	320	0.375	320	0	
exclaim2	2	2	300	0	0.006	0.500	320	0.438	320	0	

One of the nice things about Spark is that developers can use similar code for batch (Spark) and realtime (Spark Streaming) computations. Summingbird is an open source library from Twitter that offers something similar for Hadoop MapReduce and Storm: programs that look like Scala collection transformations can be executed in batch (Scalding) or realtime (Storm).

Focus on Analytics Instead of Infrastructure

Stream processing requires several components and engineering expertise[3] to setup and maintain. Available in "limited preview," a new stream processing framework from Amazon Web Services (Kinesis) eliminates[4] the need to setup stream processing infrastructure. Kinesis users can easily specify the throughput capacity they need, and shift their focus towards finding patterns and exceptions from their data streams. Kinesis integrates nicely with other popular AWS components (Redshift, S3, and Dynamodb) and should attract users already familiar with those tools. The standard AWS cost structure (no upfront fees, pay only for your usage) should also be attractive to companies who want to quickly experiment with streaming analytics.

[3] Some common components include Kafka (source of data flows), Storm or Spark Streaming (for stream data processing), and HBase / Druid / Hadoop (or some other data store).

[4] SaaS let companies focus on analytics instead of infrastructure: startups like keen.io process and analyze event data in near real-time.

Machine-Learning

In a previous post I described a few key techniques (e.g., sketches) used for *mining* data streams. Algebird is an open source abstract algebra library (used with Scalding or Summingbird) that facilitates popular stream mining computations like the Count-Min sketch and HyperLogLog. (Twitter developers observed that commutative Monoids can be used to implement[5] many popular approximation algorithms.)

Companies that specialize in analytics for machine data (Splunk, SumoLogic) incorporate machine-learning into their products. There are also general purpose software tools and web services that offer machine-learning algorithms that target streaming data. Xmp from Argyle Data includes algorithms for online learning and real-time pattern detection. FeatureStream is a new web service for applying machine-learning to data streams.

Late addition: Yahoo! recently released SAMOA—a distributed streaming machine-learning framework. SAMOA lets developers code "algorithms once and execute them in multiple" stream processing environments.

Embracing Hardware Data

Looking at the collision of hardware and software through the eyes of a data scientist

by Max Shron

In mid-May, I was at Solid, O'Reilly's new conference on the convergence of hardware and software. I went in as something close to a blank slate on the subject, as someone with (I thought) not very strong opinions about hardware in general.

[5] There's interest in developing similar libraries for Spark Streaming. A new efficient data structure for "accurate on-line accumulation of rank-based statistics" called t-digest, might be incorporated into Algebird.

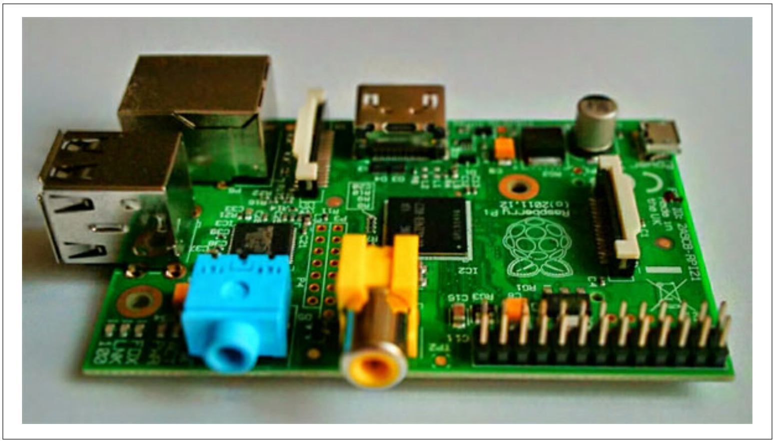


Figure 3-1. Many aspects of a hardware device can be liberally prototyped. A Raspberry Pi (such as the one seen above) can function as a temporary bridge before ARM circuit boards are put into place. Photo via Wikimedia Commons.

The talk on the grapevine in my community, data scientists who tend to deal primarily with web data, was that hardware data was the next big challenge, the place that the "alpha geeks" were heading. There are still plenty of big problems left to solve on the web, but I was curious enough to want to go check out Solid to see if I was missing out on the future. I don't have much experience with hardware—beyond wiring up LEDs as a kid, making bird houses in shop class in high school, and mucking about with an Arduino in college.

I went to Solid out of curiosity over what I would find, but also because I have spent a lot of time talking to Jon Bruner, the co-chair of Solid, and I didn't understand what he had been talking about. I've heard him talk about the "merging of hardware and software," and I've heard him say "exchange between the virtual and actual," or some variation on that, at least a few dozen times. Were these useful new concepts or nice-sounding but empty phrases?

Then there was the puzzling list of invitees to the conference. None of them seemed to fit together, apart from the fact that they dealt with tangible things. Why are cheap manufacturing, 3D printing, the Internet of Things, Fortune 50 industrial hardware companies, robotics, and consumer hardware branding exercises all in one conference? What's going on here?

After attending the conference, talking to a lot of folks, and doing some follow-up reading, I've come up with two main takeaways that are intelligible from my perspective as a software and data person trying to come to grips with what Solid is and what the trends are that it represents. First, the cost to market of hardware start-ups is reaching parity with software start-ups. And second, the material future of physical stuff is up for grabs over the next few decades. I'll cover the first in this part and the second in the follow-up article.

Cost to Market

The first major theme of Solid was that the cost of bringing a hardware start-up to market is at, or will soon reach, parity with software start-ups.

Take a moment to absorb that. Obviously, there are still great challenges in producing a hardware start-up (just as there are with a software start-up) but the cost to market is dropping, and fast. At Solid, I saw a number of consumer and industry hardware start-ups going all the way to market on low to mid six figures.

Take Drop and Birdi, two consumer-facing Internet of Things devices on display at Solid. Drop is an intelligent kitchen scale and accompanying iPad app that help make cooking recipes by weight a snap. Birdi is a smart air monitor that can discern different kinds of smoke, can detect both carbon monoxide and pollen levels, and provide alerts to your phone when its batteries are low. Both are going to market on a shoestring budget. Birdi, for example, got a $50,000 seed round from the hardware incubator Highway1, raised another $72,000 on Indiegogo, and expects to be shipping this fall.

Finding historical information on the cost of going to market for a hardware start-up is tough, but, from what I gathered at Solid, numbers in the millions or tens of millions to market used to be typical. Now, those numbers are for well-financed hardware companies instead of table stakes.

Why is that? Why has hardware gotten so much cheaper to produce than before? I can't claim to understand all the reasons, but there were many that came up in conversations at Solid. Here's what I gathered.

First, a less obvious one. Embedding computing used to mean special dialects of C written for embedded systems, or Verilog for describing complicated integrated circuits. More and more, embedded computing means a real CPU, with substantial amounts of memory. Vanilla subsets of C++ can be used for the most numerically intensive things, but interpreted languages, such as Python, Ruby, and JavaScript represent viable paths to embedded software development. I asked around while at Solid, and everyone I spoke with had coded their hardware logic in a "normal" language.

Perhaps more importantly on the software side, many aspects of a hardware device can be liberally prototyped in software. The software on the hardware can be written and iterated on using typical software development practices; complex logic and interactivity can be iterated in browser-based mockups; CPUs can be emulated before a single circuit board is created; and when a prototype gets built, Raspberry Pis and other embedded CPUs can function as temporary bridges before real ARM circuit boards get put into place.

Computing is also being split across devices. Most consumer- or industrial-facing hardware I saw at Solid consisted of the hardware itself, software on the hardware, and a phone app that provided most of the user experience. That means that all of the advances in the last decade in producing mobile software apps can be directly applied to simplifying the design and improving the user experience of hardware devices.

As a data scientist, these are some of the most exciting changes to the hardware space. I don't know much about hardware, but I do know a thing or two about deploying software at scale to add intelligence to products. As more and more of the intelligence in our hardware moves into common languages running on commodity virtual machines, opportunities will continue to open up for data scientists.

Reducing the user interface on the devices also means reduced manufacturing complexity. A common statement I heard at Solid was that every feature you added to a piece of hardware doubled the complexity to manufacture. How much simpler then is a piece of hardware when it has only one button and no complicated display? As hardware interfaces are moving onto mobile devices, the benefits are twofold: faster, cheaper iteration on the user experience, and faster, cheaper manufacture of the hardware itself. Yet another example of "software eating the world."

And, of course, the most talked about reason for reduced hardware cost: physical prototyping has gotten easier. Additive 3D printing is the best known case, but desktop cutting, laser cutting, and selective laser sintering are also greatly reducing the complexity of building prototypes.

Before I went to Solid, I wasn't aware that, traditionally, the path from CAD model to physical reality had to pass through a number of stages that oftentimes required a number of translations. First, the CAD model had to be recast as a series of negatives to cut out, then the negatives had to be translated into a tool path for the actual CNC machine, then the coordinates had to be transformed again and often manually (or via floppy disk!) entered into the computer-controlled cutting machine. Each step represents a lot of time and complexity in translation.

By contrast, the 3D printing path from CAD model to physical prototype is much more direct. Design, press go, and come back in a few hours. Desktop cutting and milling machines are getting progressively easier to use as well, reducing the time to cut out and machine complex parts.

Maker spaces, fab labs, and well-equipped university labs are putting more and better prototyping hardware within reach of inventors as well, further reducing the cost to iterate on an idea. Electronics prototyping also looks like it's getting easier (for example, LittleBits), though I don't know how much these kinds of tools are being used.

Finally, money and mentorship itself is getting easier to come by. There are now hardware accelerators (like Highway1) that both supply seed capital and connect start-ups with supply chain management in China. Kickstarter, Indiegogo, and dedicated consumer hardware suppliers like The Blueprint are bringing down the cost of capital and helping shape networks of sophisticated inventors.

This—the falling cost of going to market for hardware start-ups—is the "merging of hardware and software." It might be more precise to say that hardware, both in development and manufacturing (which has been software-driven for a long while), is becoming more and more about software over time. Thus, hardware is sharing more and more of software's strengths, including cheapness and rapid turnaround time.

Where does data fit into all of this? The theory is that cheap, ubiquitous devices will mean an even bigger explosion in data waiting to be analyzed and acted upon. It was hard to get a bead though on what the timeline is for that and what unique challenges that will pose.

Looking further ahead, Charlie Stross has pointed out that, in the next few decades, prices for embedding computing are likely to fall low enough that even adding fairly sophisticated computers to blocks of concrete won't raise their prices by much.

One nice thing about web data is that, from a machine learning perspective, e-commerce and other clickstream event data is fairly straightforward.

Sure, there are some time-series, but I haven't had to do any digital signal processing or modeling of physical systems in my time as a data scientist. Most machine learning models I have seen in practice assume a fairly high degree of independence between data points that just isn't true in the physical world.

Nor have I had to deal with power constraints, and while full-fledged embedded CPUs are now ubiquitous, don't expect to see a Hadoop cluster on your Raspberry Pi anytime soon. I'm starting to think about data flow architectures and how sensor and other kinds of data can play together. I expect it will be a useful skill a few years hence.

Editor's note: This is part one of a two-part series reflecting on the O'Reilly Solid Conference from the perspective of a data scientist. Be sure to read Max's follow-up on truly digital manufacturing.

Extracting Value from the IoT

Data from the Internet of Things makes an integrated data strategy vital

by Mike Barlow

The Internet of Things (IoT) is more than a network of smart toasters, refrigerators, and thermostats. For the moment, though, domestic appliances are the most visible aspect of the IoT. But they represent merely the tip of a very large and mostly invisible iceberg.

IDC predicts by the end of 2020, the IoT will encompass 212 billion "things," including hardware we tend not to think about: compressors, pumps, generators, turbines, blowers, rotary kilns, oil-drilling equipment, conveyer belts, diesel locomotives, and medical imaging scanners, to name a few. Sensors embedded in such machines and devices use the IoT to transmit data on such metrics as vibration, temperature, humidity, wind speed, location, fuel consumption, radiation levels, and hundreds of other variables.

Figure 3-2. Union Pacific uses infrared and audio sensors placed on its tracks to gauge the state of wheels and bearings as the trains pass by. Photo by Rick Cooper, on Wikimedia Commons.

"Machines can be very chatty," says William Ruh, a vice president and corporate officer at GE.

Ruh's current focus is to drive the company's efforts to develop an "industrial" Internet that blends three elements: intelligent machines, advanced analytics, and empowered users. Together, those elements generate a variety of data at a rapid pace, creating a deluge that makes early definitions of big data seem wildly understated.

Making sense of that data and using it to produce a steady stream of usable insights require infrastructure and processes that are fast, accurate, reliable, and scalable. Merely collecting data and loading it into a data warehouse is not sufficient—you also need capabilities for accessing, modeling, and analyzing your data; a system for sharing results across a network of stakeholders; and a culture that supports and encourages real-time collaboration.

What you *don't* need is a patchwork of independent data silos in which information is stockpiled like tons of surplus grain. What you *do* need are industrial-grade, integrated processes for managing and extracting value from IoT data and traditional sources.

Dan Graham, general manager for enterprise systems at Teradata, sees two distinct areas in which integrated data will create significant business value: product development and product deployment.

"In the R&D or development phase, you will use integrated data to see how all the moving parts will work together and how they interact. You can see where the friction exists. You're not looking at parts in isolation. You can see the parts within the context of your supply chain, inventory, sales, market demand, channel partners, and many other factors," says Graham.

The second phase is post-sales deployment. "Now you use your integrated data for condition-based (predictive) maintenance. Airplanes, locomotives, earth movers, automobiles, disk drives, ATMs, and cash registers require continual care and support. Parts wear out and fail. It's good to know which parts from which vendors fail, how often they fail, and the conditions in which they fail. Then you can take the device or machine offline and repair it before it breaks down," says Graham.

For example, microscopic changes in the circumference of a wheel or too little grease on the axle of a railroad car, can result in delays and even derailments of high-speed freight trains. Union Pacific, the largest railroad company in the US, uses a sophisticated system of sensors and analytics to predict when critical parts are likely to fail, enabling maintenance crews to fix problems while rolling stock is in the rail yard. The alternative, which is both dangerous and expensive, would be waiting for parts to fail while the trains are running.

Union Pacific uses infrared and audio sensors placed on its tracks to gauge the state of wheels and bearings as the trains pass by. It also uses ultrasound to spot flaws or damage in critical components that could lead to problems. On an average day, the railroad collects 20 million sensor readings from 3,350 trains and 32,000 miles of track. It then uses pattern-matching algorithms to detect potential issues and flag them for action. The effort is already paying off: Union Pacific has cut bearing-related derailments by 75%, says Graham.[6]

NCR Corporation, which pioneered the mechanical cash register in the 19th century, is currently the global leader in consumer transaction technologies. The company provides software, hardware, and

[6] Murphy, Chris. "High-Speed Analytics: Union Pacific shows the potential of the instrumented, interconnected, analytics-intensive enterprise." *Information Week*, August 13, 2012.

services, enabling more than 485 million transactions daily at large and small organizations in retail, financial, travel, hospitality, telecom, and technology sectors. NCR gathers data telemetrically from the IoT—data generated by ATMs, kiosks, point-of-sale terminals, and self-service checkout machines handling a total of about 3,500 transactions per second. NCR then applies its own custom algorithms to predict which of those devices is likely to fail and to make sure the right technician, with the right part, reaches the right location before the failure occurs.

Underneath the hood of NCR's big data/IoT strategy is a unified data architecture that combines an integrated data warehouse, Hadoop, and the Teradata Aster Discovery Platform. The key operating principle is integration, which assures that data flowing in from the IoT is analyzed in context with data from multiple sources.

"The name of the game is exogenous data," says Michael Minelli, an executive at MasterCard and co-author of *Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses*. "You need the capabilities and skills for combining and analyzing data from various sources that are outside the four walls of your organization. Then you need to convert data into actionable insights that will drive better decisions and grow your business. Data from the IoT is just one of many external sources you need to manage in combination with the data you already own."

From Minelli's perspective, data from the IoT is additive and complementary to the data in your data warehouse. Harvey Koeppel, former CIO at Citigroup Global Consumer Banking, agrees. "The reality is that there is still a legacy environment, and it's not going away anytime soon. Facts are facts; they need to be collected, stored, organized, and maintained. That's certainly the case for Fortune 1000 companies, and I expect it will remain that way for the foreseeable future," says Koeppel.

Big data collected from the IoT tends to be "more ephemeral" than traditional types of data, says Koeppel. "Geospatial data gathered for a marketing campaign is different than financial data stored in your company's book of record. Data that's used to generate a coupon on your mobile phone is not in the same class as data you're required to store because of a government regulation."

That said, big data from the IoT is rapidly losing its status as a special case or oddity. With each passing day, big data is perceived as just another item on the menu. Ideally, your data architecture and data warehouse systems would enable you to work with whichever type of data you need, whenever you need it, to create actionable insights that lead to improved outcomes across a variety of possible activities.

"In the best of all worlds, we would blend data from the IoT with data in the data warehouse to create the best possible offers for consumers in real time or to let you know that your car is going to run out of gas 10 minutes from now," says Koeppel. "The thoughtful approach is combining data from a continuum of sources, ranging from the IoT to the traditional data warehouse."

This post is part of a collaboration between O'Reilly and Teradata exploring the convergence of hardware and software. See our statement of editorial independence.

Fast Data Calls for New Ways to Manage Its Flow

Examples of multi-layer, three-tier data-processing architecture

by Andy Oram

Like CPU caches, which tend to be arranged in multiple levels, modern organizations direct their data into different data stores under the principle that a small amount is needed for real-time decisions and the rest for long-range business decisions. This article looks at options for data storage, focusing on one that's particularly appropriate for the "fast data" scenario described in a recent O'Reilly report.

Many organizations deal with data on at least three levels:

1. They need data at their fingertips, rather like a reference book you leave on your desk. Organizations use such data for things like determining which ad to display on a web page, what kind of deal to offer a visitor to their website, or what email message to suppress as spam. They store such data in memory, often in key/value stores that allow fast lookups. Flash is a second layer (slower than memory, but much cheaper), as I described in a recent article. John Piekos, vice president of engineering at

VoltDB, which makes an in-memory database, says that this type of data storage is used in situations where delays of just 20 or 30 milliseconds mean lost business.

2. For business intelligence, these organizations use a traditional relational database or a more modern "big data" tool such as Hadoop or Spark. Although the use of a relational database for background processing is generally called online analytic processing (OLAP), it is nowhere near as online as the previous data used over a period of just milliseconds for real-time decisions.
3. Some data is archived with no immediate use in mind. It can be compressed and perhaps even stored on magnetic tape.

For the new fast data tier, where performance is critical, techniques such as materialized views further improve responsiveness. According to Piekos, materialized views bypass a certain amount of database processing to cut milliseconds off of queries. Materialized views can be compared to a column in a spreadsheet that is based on a calculation using other columns and is updated as the spreadsheet itself is updated. In a database, an SQL query defines a materialized view. As rows are inserted, deleted, or modified in the underlying table on which the view is based, the materialized view calculations are automatically updated. Naturally, the users must decide in advance what computation is crucial to them and define the queries accordingly. The result is an effectively instant, updated result suitable for real-time decision-making.

Some examples of the multi-layer, three-tier architecture cited by Piekos are:

- Ericsson, the well-known telecom company, puts out a set-top box for television viewers. These boxes collect a huge amount of information about channel changes and data transfers, potentially from hundreds of millions of viewers. Therefore, one of the challenges is just writing to the database at a rate that supports the volume of data they receive. They store the data in the cloud, where they count such things as latency, response time, and error rates. Data is then pushed to a slower, historical data store.
- Sakura, one of the largest Japanese ISPs, uses their database for protection against distributed Denial of Service attacks. This requires quick recognition of a spike in incoming network traf-

fic, plus the ability to distinguish anomalies from regular network uses. Every IP packet transmitted through Sakura is logged to a VoltDB database—but only for two or three hours, after which the traffic is discarded to make room for more. The result is much more subtle than traditional blacklists, which punish innocent network users who happen to share a block of IP addresses with the attacker.

- Flytxt, which analyzes telecom messages for communication service providers, extracts intelligence from four billion events per day, streaming from more than 200 million mobile subscribers. With this data, operators can make quick decisions, such as whether the customer's balance covers the call, whether the customer is a minor whose call should be blocked through parental controls, and so forth. This requires complex SQL queries, which a materialized view enables in the short time desired.

The choices for data storage these days are nearly overwhelming. There are currently no clear winners—each option has value in particular situations. Therefore, we should not be surprised that users are taking advantage of two or more solutions at once, each for what it is best at doing.

This post is part of a collaboration between O'Reilly and VoltDB exploring fast and big data. See our statement of editorial independence.

Clouds, Edges, Fog, and the Pendulum of Distributed Computing

by Alistair Croll

The history of computing has been a constant pendulum, swinging between centralization and distribution.

The first computers filled rooms, and operators were physically within them, switching toggles and turning wheels. Then came mainframes, which were centralized, with dumb terminals.

As the cost of computing dropped and the applications became more democratized, user interfaces mattered more. The smarter clients at the edge became the first personal computers; many broke free of the network entirely. The client got the glory; the server merely handled queries.

Once the web arrived, we centralized again. LAMP (Linux, Apache, MySQL, PHP) buried deep inside data centers, with the computer at the other end of the connection relegated to little more than a smart terminal rendering HTML. Load-balancers sprayed traffic across thousands of cheap machines. Eventually, the web turned from static sites to complex software as a service (SaaS) applications.

Then the pendulum swung back to the edge, and the clients got smart again. First with AJAX, Java, and Flash; then in the form of mobile apps where the smartphone or tablet did most of the hard work and the back-end was a communications channel for reporting the results of local action.

Now we're seeing the first iteration of the Internet of Things (IoT), in which small devices, sipping from their batteries, chatting carefully over Bluetooth LE, are little more than sensors. The preponderance of the work, from data cleaning to aggregation to analysis, has once again moved to the core: the first versions of the Jawbone Up band doesn't do much until they send their data to the cloud.

But already we can see how the pendulum will swing back. There's a renewed interest in computing at the edges—Cisco calls it "fog computing": small, local clouds that combine tiny sensors with more powerful local computing—and this may move much of the work out to the device or the local network again. Companies like realm.io are building databases that can run on smartphones or even wearables. Foghorn Systems is building platforms on which developers can deploy such multi-tiered architectures. Resin.io calls this "strong devices, weakly connected".

Systems architects understand well the tension between putting everything at the core, and making the edges more important. Centralization gives us power, makes managing changes consistent and easy, and cuts on costly latency and networking; distribution gives us more compelling user experiences, better protection against central outages or catastrophic failures, and a tiered hierarchy of processing that can scale better. Ultimately, each swing of the pendulum gives us new architectures and new bottlenecks; each rung we climb up the stack brings both abstraction and efficiency.

Data (Science) Pipelines

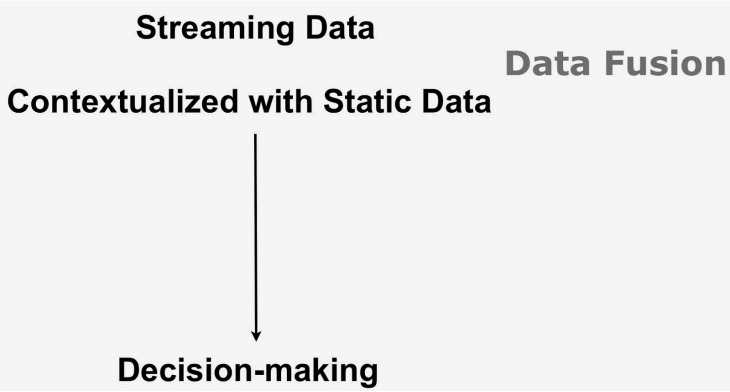
This chapter tackles the nitty-gritty of data work—not on the data side but the data scientist's side. Ben Lorica tackles the combinations of tools available off-the-shelf (and the platforms that enable combining tools), as well as the process of feature discovery and selection.

Verticalized Big Data Solutions

General-purpose platforms can come across as hammers in search of nails

by Ben Lorica

As much as I love talking about general-purpose big data platforms and data science frameworks, I'm the first to admit that many of the interesting startups I talk to are focused on specific verticals. At their core, big data *applications* merge large amounts of real-time and static data to improve decision-making:



This simple idea can be hard to execute in practice (think volume, variety, velocity). Unlocking value from disparate data sources entails some familiarity with *domain-specific*[1] data sources, requirements, and business problems.

It's difficult enough to solve a specific problem, let alone a generic one. Consider the case of Guavus—a successful startup that builds big data solutions for the *telecom* industry ("communication service providers"). Its founder[2] was very familiar with the data sources in telecom, and knew the types of applications that would resonate within that industry. Once they solve one set of problems for a telecom company (network optimization), they quickly leverage the same systems to solve others (marketing analytics).

This ability to address a variety of problems stems from Guavus' deep familiarity with data and problems in telecom. In contrast, a typical general-purpose platform can come across as a hammer in search of a nail. So while I remain a fan (and user) of general-purpose platforms, the less well-known verticalized solutions are definitely on my radar.

[1] General-purpose platforms and components are helpful, but they usually need to be "tweaked" or "optimized" to solve problems in a variety of domains.

[2] This post grew out of a recent conversation with Guavus founder, Anukool Lakhina.

Better Tools Can't Overcome Poor Analysis

I'm not suggesting that the criticisms raised against big data don't apply to verticalized solutions. But many problems are due to poor analysis and not the underlying tools. A few of the more common criticisms arise from analyzing correlations: correlation is not causation, correlations are dynamic and can sometimes change drastically,[3] and data dredging.[4]

Scaling Up Data Frames

New frameworks for interactive business analysis and advanced analytics fuel the rise in tabular data objects

by Ben Lorica

Long before the advent of "big data," analysts were building models using tools like R (and its forerunners S/S-PLUS). Productivity hinged on tools that made data wrangling, data inspection, and data modeling convenient. Among R users, this meant proficiency with data frames—objects used to store *data matrices* that can hold both numeric and categorical data. A `data.frame` is the data structure consumed by most R analytic libraries.

But not all data scientists use R, nor is R suitable for all data problems. I've been watching with interest the growing number of alternative data structures for business analysis and advanced analytics. These new tools are designed to handle much larger data sets and are frequently optimized for specific problems. And they all use idioms that are familiar to data scientists—either SQL-like expressions, or syntax similar to those used for R `data.frame` or `pandas.DataFrame`.

As much as I'd like these different projects and tools to coalesce, there are differences in the platforms they inhabit, the use cases they target, and the (business) objectives of their creators. Regardless of

[3] When I started working as a quant at a hedge fund, traders always warned me that correlations jump to 1 during market panics.

[4] The best example comes from finance and involves the S&P 500 and butter production in Bangladesh.

their specific features and goals, these emerging tools[5] and projects all need data structures that simplify data munging and *data analysis* —including data alignment, how to handle missing values, standardizing values, and coding categorical variables.

Spark

As *the* data processing engine for big data, analytic libraries and features are making their way into Spark,[6] thus objects and data structures that simplify data wrangling and analysis are also beginning to appear. For advanced analytics and machine learning, MLTable is a table-like interface that mimics structures like R `data.frame`, database tables, or MATLAB's dataset array. For *business analytics* (interactive query analysis), SchemaRDD's are used in relational queries executed in Spark SQL.

ADATAO
Data intelligence for all

At the recent Spark Summit, start-up Adatao unveiled and announced plans to open sourceDistributed Data Frames (DDF)—objects that were heavily inspired by R `data.frame`. Adatao developed DDF as part of their pAnalytics and pInsights products, so DDF comes with many utilities for analysis and data wrangling.

R

Inspired by idioms used for R `data.frame`, Adatao's DDF can be used from within RStudio. With standard R code,[7] users can access a

[5] For this short piece, I'm skipping the many tabular data structures and columnar storage projects in the Hadoop ecosystem, and I'm focusing on the new tools that target (or were created by) data scientists.

[6] Full disclosure: I am an advisor to Databricks—a start-up commercializing Apache Spark.

[7] DDF is an ambitious project that aims to simplify big data analytics for users across languages and compute engines. It can be accessed using other languages including Python, Scala, and Java. It is also designed for multiple engines. In a demo, data from an HBase table is read into a DDF, data cleansing and machine learning operations are performed on it using Spark, and results are written back out to S3, all using DDF idioms.

collection of highly scalable analytic libraries (the algorithms are executed in Spark).

```
ddf <- adatao.getDDF("ddf://adatao/flightInfo")
adatao.setMutable(ddf, TRUE)
adatao.dropNA(ddf)
adatao.transform(ddf, "delayed = if(arrdelay > 15, 1, 0)")
# adatao implementation of lm
model  <-  adatao.lm(delayed  ~  distance  +  deptime  +  depdelay,
data=ddf)
lmpred <- adatao.predict(model, ddf1)
```

For interactive queries, new R packages dplyr and/or data.table can be used for fast aggregations and joins. dplyr also comes with an operator (%.%) for chaining together data (wrangling) operations.

Python

Among data scientists who use Python, pandas.DataFrame has been an essential tool ever since its release. Over the past few years pandas has become one of the most active open source projects in the data space (266 distinct contributors and counting). But pandas was designed for small to medium sized data sets, and as pandas creator Wes McKinney recently noted, there are many areas for improvement.

One area is scalability. To scale to terabytes of data, a new alternative is GraphLab's SFrame, a component of a product called GraphLab Create. GraphLab Create targets Python users: it comes with a Python API and detailed examples contained in IPython notebooks. SFrame itself uses syntax that should be easy for pandas users to pick up. There are plans to open source SFrame (and some other components of GraphLab Create) later this year.

```
# recommender in five lines of Python
import graphlab
data = graphlab.SFrame("s3://my_bucket/my_data.csv")
model = graphlab.recommender.create(data)
model.recommend(data)
model.save("s3://my_bucket/my_model.gl")
```

Badger

Badger is a new tabular analytics library being built at DataPad—a start-up led and co-founded by Wes McKinney.

A C library coupled with a Python-based interface, Badger targets “business analytics and BI use cases” and has a pandas-like syntax, designed for data processing and analytical queries (“more expressive than SQL”). As an in-memory query processor, it features active memory management and caching, and targets interactive speeds on 100-million row and smaller data sets on single machines.

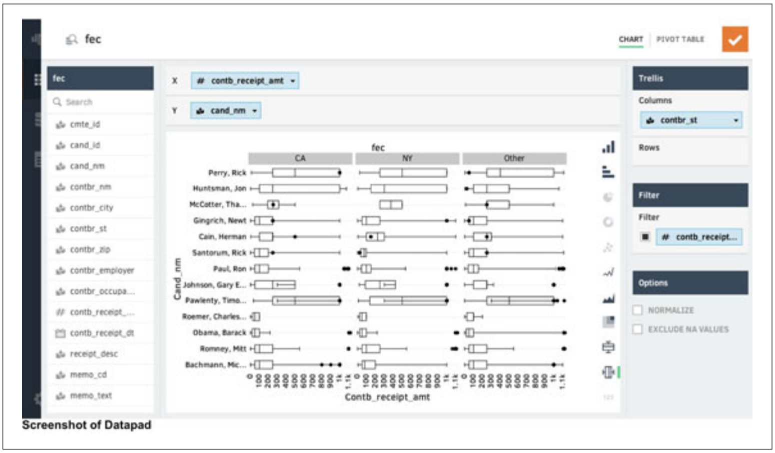


Figure 4-1. Screenshot of DataPad

Badger is currently only available as part of DataPad’s visual analysis platform. But its *lineage* (developed by the team that created pandas) combined with promising performance reports have many Pydata users itching to try it out.

Streamlining Feature Engineering

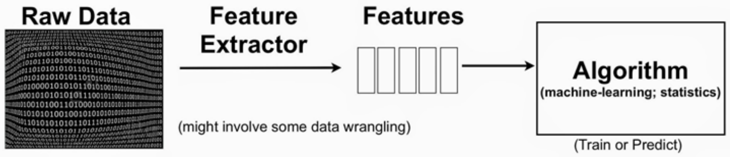
Researchers and startups are building tools that enable feature discovery

by Ben Lorica

Why do data scientists spend so much time on data wrangling and data preparation? In many cases it's because they want access to the best *variables* with which to build their models. These *variables* are known as *features* in machine-learning parlance. For many[8] data applications, feature engineering and feature selection are just as (if not more important) than choice of algorithm:

> Good features allow a simple model to beat a complex model (to paraphrase Alon Halevy, Peter Norvig, and Fernando Pereira).

The terminology can be a bit confusing, but to put things in context one can simplify the data science pipeline to highlight the importance of features:


Raw Data
Feature Extractor
Features
Algorithm
(machine-learning; statistics)
(might involve some data wrangling)
(Train or Predict)

Feature Engineering or the Creation of New Features

A simple example to keep in mind is text mining. One starts with *raw* text (documents) and extracted features could be individual words or phrases. In this setting, a feature could indicate the frequency of a specific word or phrase. Features[9] are then used to classify and cluster documents, or extract topics associated with the raw text. The process usually involves the creation[10] of new features (fea-

[8] The quote from Alon Halevy, Peter Norvig, and Fernando Pereira is associated with *big* data. But features are just as important in small data problems. Read through the Kaggle blog and you quickly realize that winning entries spend a lot of their time on feature engineering.

[9] In the process documents usually get converted into structures that algorithms can handle (vectors).

[10] Once can for example create composite (e.g. linear combination) features out of existing ones.

ture *engineering*) and identifying the most essential ones (feature *selection*).

Feature Selection Techniques

Why bother selecting features? Why not use all available features? Part of the answer could be that you need a solution that is simple, interpretable, and fast. This favors features that have good statistical performance and that are easy to explain to non-technical users. But there could be legal[11] reasons for excluding certain features as well (e.g., the use of credit scores is discriminatory in certain situations).

In the machine-learning literature there are three commonly used methods for feature selection:

- Domain experts can manually pick out features, and more recently I wrote about a service that uses crowdsourcing techniques. It's not hard to find examples of problems where domain expertise is insufficient, and this approach isn't particularly practical when underlying data sets are massive.
- There are variable ranking procedures that use metrics like correlation, information criteria, etc. They scale to large data sets but can easily lead to *strange recommendations* (e.g., use "butter production in Bangladesh" to predict the S&P 500).
- Techniques that take a vast feature space and reduce it to a lower-dimensional one (clustering, principal component analysis, matrix factorization).

Expect More Tools to Streamline Feature Discovery

In practice, feature selection and feature engineering are iterative processes where humans leverage automation[12] to wade through candidate features. Statistical software have long had (stepwise) procedures for feature selection. New startups are providing similar tools: Skytree's new user interface lets *business* users automate feature selection.

[11] From Materialization Optimizations for Feature Selection Workloads: *"Using credit score as a feature is considered a discriminatory practice by the insurance commissions in both California and Massachusetts."*

[12] Stepwise procedures in statistical regression is a familiar example.

I'm definitely noticing much more interest from researchers and startups. A group out of Stanford[13] just released a paper on a new R language extension and execution framework designed for feature selection. Their R extension enables data analysts to incorporate feature selection using high-level constructs that form a domain specific language. Some startups like ContextRelevant and SparkBeyond,[14] are working to provide users with tools that simplify feature engineering and selection. In some instances this includes incorporating features derived from external data sources. Users of SparkBeyond are able to incorporate the company's knowledge databases (Wikipedia, OpenStreeMap, Github, etc.) to enrich their own data sources.

While many startups who build analytic tools begin by focusing on algorithms, many products will soon begin highlighting how they handle feature selection and discovery. There are many reasons why there will be more emphasis on features: *interpretability* (this includes finding *actionable* features that drive model performance), *big data* (companies have many more data sources to draw upon), and an appreciation of *data pipelines* (algorithms are just one component).

Big Data Solutions Through the Combination of Tools

Applications get easier to build as packaged combinations of open source tools become available

by Ben Lorica

As a user who tends to mix-and-match many different tools, not having to deal with configuring and assembling a suite of tools is a big win. So I'm really liking the recent trend towards more integrated and packaged solutions. A recent example is the relaunch of Cloudera's Enterprise Data hub, to include Spark[15] *and* Spark Streaming. Users benefit by gaining automatic access to analytic

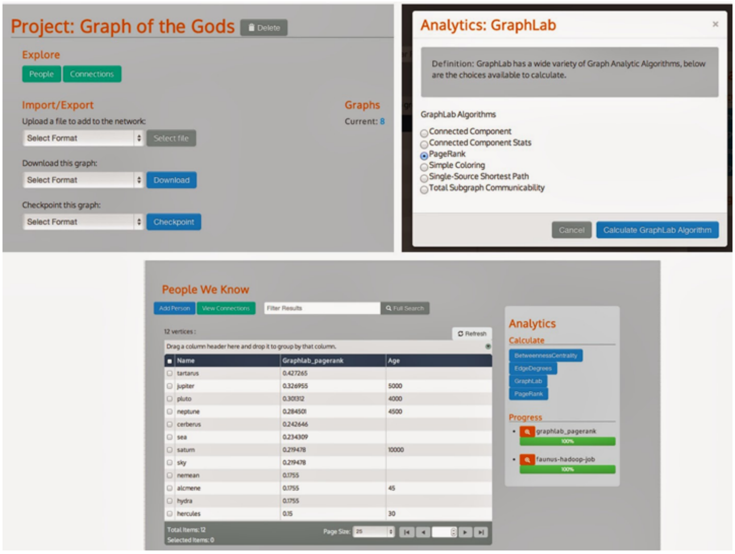
[13] The Stanford research team designed their feature selection tool after talking to data analysts at several companies. The goal of their project was to increase analyst productivity.

[14] Full disclosure: I'm an advisor to SparkBeyond.

[15] Full disclosure: I am an advisor to Databricks—a startup commercializing Spark.

engines that come with Spark.[16] Besides simplifying things for data scientists and data engineers, easy access to analytic engines is critical for streamlining the creation of big data *applications*.

Another recent example is Dendrite[17]—an interesting new graph analysis solution from Lab41. It combines Titan (a *distributed* graph database), GraphLab (for graph analytics), and a front-end that leverages AngularJS, into a Graph exploration and analysis tool for business analysts:



Graphs

Current: 8

Upload a file to add to the network: Select Format

Download this graph: Select Format

Checkpoint this graph: Select Format

Definition: GraphLab has a wide variety of Graph Analytic Algorithms, below are the choices available to calculate.

GraphLab Algorithms

○ Connected Component
○ Connected Component Stats
● PageRank
○ Simple Coloring
○ Single-Source Shortest Path
○ Total Subgraph Communicability

12 vertices :

Drag a column header here and drop it to group by that column.

□	Name	Graphlab_pagerank	Age
□	tartarus	0.427265	
□	jupiter	0.326955	5000
□	pluto	0.301312	4000
□	neptune	0.284501	4500
□	cerberus	0.242646	
□	sea	0.234309	
□	saturn	0.219478	10000
□	sky	0.219478	
□	nemean	0.1755	
□	alcmene	0.1755	45
□	hydra	0.1755	
□	hercules	0.15	30

Total Items: 12
Selected Items: 0
Page Size: 25

- graphlab_pagerank 100%
- faunus-hadoop-job 100%

Users of Spark explore Spark Streaming because similar code for batch (Spark) can, with minor modification, be used for realtime (Spark Streaming) computations. Along these lines, Summingbird—an open source library from Twitter—offers something similar for Hadoop MapReduce and Storm. With Summingbird, programs that look like Scala collection transformations can be executed in batch (Scalding) or realtime (Storm).

In some instances the underlying techniques from a set of tools makes its way into others. The DeepDive team at Stanford just

[16] Some potential applications of Spark and Spark Streaming include stream processing and mining, interactive and iterative computing, machine-learning, and graph analytics.

[17] Hat tip to Danny Bickson.

recently revamped their information extraction and natural language understanding system. But already techniques used in DeepDive have found their way into many other systems including MADlib, Cloudera Impala, “a product from Oracle,” and Google Brain.

The Evolving, Maturing Marketplace of Big Data Components

As the last chapter looked at data from the data scientist's point of view, this chapter explores data science from the other side: the hardware and software that store, sort, and operate on the 1's and 0's. Andy Oram's exploration of Flash and its impact on databases leads off: the benefits to be reaped from solid-state memory are much greater when baked in, rather than simply swapping in flash drives for spinning magnetic media. Following sections tackle Hadoop 2.0 (a notable release this past year), the growth of Spark, and the provocative proposition that "the data center needs an operating system."

How Flash changes the design of database storage engines

High-performing memory throws many traditional decisions overboard

by Andy Oram

Over the past decade, SSD drives (popularly known as Flash) have radically changed computing at both the consumer level—where USB sticks have effectively replaced CDs for transporting files—and the server level, where it offers a price/performance ratio radically different from both RAM and disk drives. But databases have just started to catch up during the past few years. Most still depend on internal data structures and storage management fine-tuned for spinning disks.

Citing price and performance, one author advised a wide range of database vendors to move to Flash. Certainly, a database administrator can speed up old databases just by swapping out disk drives and inserting Flash, but doing so captures just a sliver of the potential performance improvement promised by Flash. For this article, I asked several database experts—including representatives of Aerospike, Cassandra, FoundationDB, RethinkDB, and Tokutek—how Flash changes the design of storage engines for databases. The various ways these companies have responded to its promise in their database designs are instructive to readers designing applications and looking for the best storage solutions.

It's worth noting that most of the products discussed here would fit roughly into the category known as NoSQL, but that Tokutek's storage engine runs on MySQL, MariaDB, and Percona Server, as well as MongoDB; the RethinkDB engine was originally developed for MySQL; and many of the tools I cover support various features, such as transactions that are commonly associated with relational databases. But this article is not a description of any database product, much less a comparison—it is an exploration of Flash and its impact on databases.

Key Characteristics of Flash that Influence Databases

We all know a few special traits of Flash—that it is fast, that its blocks wear out after a certain number of writes—but its details can have a profound effect on the performance of databases and applications. Furthermore, discussions of speed should be divided into throughput and latency, as with networks. The characteristics I talked to the database experts about included:

Random reads

Like main memory, but unlike traditional disks, Flash serves up data equally fast, no matter how much physical distance lies between the reads. However, one has to read a whole block at a time, so applications may still benefit from locality of reference because a read that is close to an earlier read may be satisfied by main memory or the cache.

Throughput

Raw throughput of hundreds of thousands of reads or writes per second has been recorded. Many tools take advantage of this advantage—two orders of magnitude better than disks, or more. And throughput continues to improve as density is improving, according to Aerospike CTO Brian Bulkowski, because there are more blocks per chip at higher densities, leading to higher throughput.

Latency

According to David Rosenthal, CEO of FoundationDB, read latency is usually around 50 to 100 microseconds. As pointed out by Slava Akhmechetat, CEO of RethinkDB, Flash is at least a hundred times faster than disks, which tend more toward 5-50 milliseconds per read. Like CPU speeds, however, Flash seems to have reached its limit in latency and is not improving.

Parallelism

Flash drives offer multiple controllers (originally nine, and now usually more), or single higher-performance controllers. This rewards database designs that can use multiple threads and cores, and split up workloads into many independent reads or writes.

At the risk of oversimplifying, you could think of Flash as combining the convenient random access of main memory with the size and durability of disk, having access speeds between the two. Ideally, for performance, an application would run all in memory, and several database solutions were created as in-memory solutions (VoltDB, Oracle Times Ten, and the original version of MySQL Cluster known as NDB Cluster, come to mind). But as data sets grow, memory has become too expensive, and Flash arises as an attractive alternative.

The Order of the Day

Several of the products I researched replace the file system with their own storage algorithms, as many relational databases have also done. Locality, as we've seen, can affect the speed of reads. But it isn't crucial to strive for locality of data. The overhead of writing data in the "right" spot may override the trivial overhead of doing multiple reads when Flash makes throughput so high.

Aerospike is the first database product designed from the beginning with Flash in mind. Bulkowski says this choice was made when the founders noted that data too large or too expensive to fit in RAM is well suited for Flash.

Although it can run simply with RAM backed by rotational disks, Aerospike stores indexes in RAM and the rest of the data in Flash. This way, they can quickly look up the index in RAM and then retrieve data from multiple Flash drives in parallel. Because the indexes are updated in RAM, writes to Flash are greatly reduced. Monica Pal, chief marketing officer for Aerospike, comments that the new category of real-time, big data driven applications, striving to personalize the user experience, have a random read/write data access pattern. Therefore, many customers use Aerospike to replace the caching tier at a site.

Of the databases covered in this article, Cassandra is perhaps the most explicit in ordering data to achieve locality of reference. Its fundamental data structure is a log-structured merge-tree (LSM-tree), very different from the B-tree family found in most relational and NoSQL databases. An LSM-tree tries to keep data sorted into ordered sets, and regularly combines sets to make them as long as possible. Developed originally to minimize extra seeks on disk, LSM-trees later proved useful with Flash to dramatically reduce writes (although random reads become less efficient).

According to Jonathan Ellis, project chair of Apache Cassandra, the database takes on a lot of the defragmentation operations that most applications leave up to the file system, in order to maintain its LSM-trees efficiently. It also takes advantage of its knowledge of the data (for instance, which blocks belong to a single BLOB) to minimize the amount of garbage collection necessary.

Rosenthal, in contrast, says that the FoundationDB team counted on Flash controllers to solve the problem of fragmented writes. Just as sophisticated disk drives buffer and reorder writes to minimize seeks (such as with elevator algorithms), Flash drives started six or seven years ago to differentiate themselves in the market by offering controllers that could do what LSM does at the database engine level. Now, most Flash controllers offer these algorithms.

Tokutek offers a clustered database, keeping data with the index. They find clustering ideal for retrieving ranges of data, particularly when there are repeated values, as in denormalized NoSQL document stores like the MongoDB database that Tokutek supports. Compression also reduces read and write overhead, saving money on storage as well as time. Tokutek generally gets a compression ratio of 5 or 7 to 1 for MySQL and MariaDB. They do even better on MongoDB—perhaps 10 to 1 compression—because its document stores are denormalized compared to a relational database.

Write Right

Aerospike, FoundationDB, RethinkDB, and Tokutek use MVCC or a similar concept to write new versions of data continuously and clean up old versions later, instead of directly replacing stored data with new values.

In general, a single write request by a database can turn into multiple writes because of the need to update the data as well as multiples indexes. But Bulkowski says that, by storing indexes in memory, Aerospike achieves a predictable write amplification of 2, whereas other applications often suffer from a factor of 10. He points out that this design choice, like many made by Aerospike, is always being reconsidered as they research the best ways to speed up and scale applications.

He is also not worried about the fabled problem of wear in Flash. They can actually last as long as the systems' sites normally expect to last—for instance, the blocks on an Intel S3700 SSD can last five years with 10 writes per day.

Both Aerospike and FoundationDB offer strict durability guarantees. All writes go to Flash and are synched.

Keep 'Em Coming

Rosenthal says that the increased speed and parallelism of Flash—perhaps 100,000 operations per second, compared to 300 for typical disks—create the most change to database design. "The traditional design of relational databases, with one thread per connection, worked fine when disks were bottleneck," he says, "but now the threads become the bottleneck." FoundationDB internally uses its own light-weight processes, like Erlang does. Rosenthal says that the stagnation in latency improvement makes parallelism even more important.

Because of extensive parallelism, Bulkowski says that deep queues work better on Flash than rotational disks. Aerospike is particularly useful for personalization at real-time speeds, which is used by applications as diverse as ecommerce sites, web ad auctions (real-time bidding), the notoriously variable airline pricing, and even telecom routing. These applications combine a large number of small bits of information (from just gigabytes to more than 100TB) about pages, tickets, and products, as well as subscribers, customers or users, requiring a lot of little database transactions. The concurrency provided by Aerospike allows sites to complete these database operations consistently, 99% of the time, within a couple of milliseconds, and to scale cost effectively, on surprisingly small clusters.

These new database storage engine designs have clearly thrown many traditional decisions overboard. It is up to application developers now to review their database schemas and access pattern assumptions to take advantage of these developments.

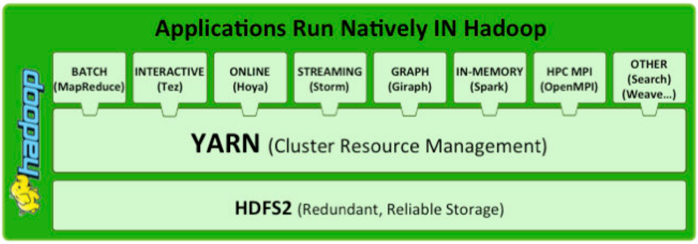
This post is part of a collaboration between O'Reilly and Aerospike. See our statement of editorial independence.

Introduction to Hadoop 2.0

by Rich Raposa

Apache Hadoop 2.0 represents a generational shift in the architecture of Apache Hadoop. With YARN, Apache Hadoop is recast as a significantly more powerful platform—one that takes Hadoop beyond merely batch applications to taking its position as a 'data operating system' where HDFS is the file system and YARN is the operating system.

YARN is a re-architecture of Hadoop that allows multiple applications to run on the same platform. With YARN, applications run "in" Hadoop, instead of "on" Hadoop:


Applications Run Natively IN Hadoop
hadoop
BATCH (MapReduce)
INTERACTIVE (Tez)
ONLINE (Hoya)
STREAMING (Storm)
GRAPH (Giraph)
IN-MEMORY (Spark)
HPC MPI (OpenMPI)
OTHER (Search) (Weave...)
YARN (Cluster Resource Management)
HDFS2 (Redundant, Reliable Storage)

The fundamental idea of YARN is to split up the two major responsibilities of the JobTracker and TaskTracker into separate entities. In Hadoop 2.0, the JobTracker and TaskTracker no longer exist and have been replaced by three components:

ResourceManager
: A scheduler that allocates available resources in the cluster amongst the competing applications.

NodeManager
: Runs on each node in the cluster and takes direction from the ResourceManager. It is responsible for managing resources available on a single node.

ApplicationMaster
: An instance of a framework-specific library, an ApplicationMaster runs a specific YARN job and is responsible for negotiating resources from the ResourceManager and also working with the NodeManager to execute and monitor Containers.

The actual data processing occurs within the Containers executed by the ApplicationMaster. A Container grants rights to an application to use a specific amount of resources (memory, CPU, etc.) on a specific host.

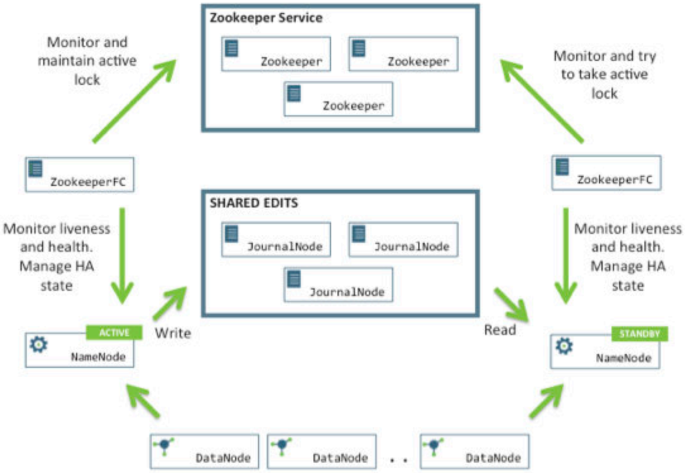
YARN is not the only new major feature of Hadoop 2.0. HDFS has undergone a major transformation with a collection of new features that include:

NameNode HA
: Automated failover with a hot standby and resiliency for the NameNode master service.

Snapshots
: Point-in-time recovery for backup, disaster recovery and protection against use errors.

Federation
: A clear separation of namespace and storage by enabling generic block storage layer.

NameNode HA is achieved using existing components like ZooKeeper along with new components like a quorum of JournalNodes and the ZooKeeper Failover Controller (ZKFC) processes:



Federation enables support for multiple namespaces in the cluster to improve scalability and isolation. Federation also opens up the architecture, expanding the applicability of HDFS cluster to new implementations and use cases.

In our tutorial at Strata 2014, we discussed the details of YARN and provided an overview of how you might develop your own YARN implementation. We also discussed the components of HDFS High Availability, how to protect your enterprise data with HDFS Snapshots, and how Federation can be used to utilize your cluster resources more effectively. We also included a brief discussion on migrating from Hadoop 1.x to 2.0.

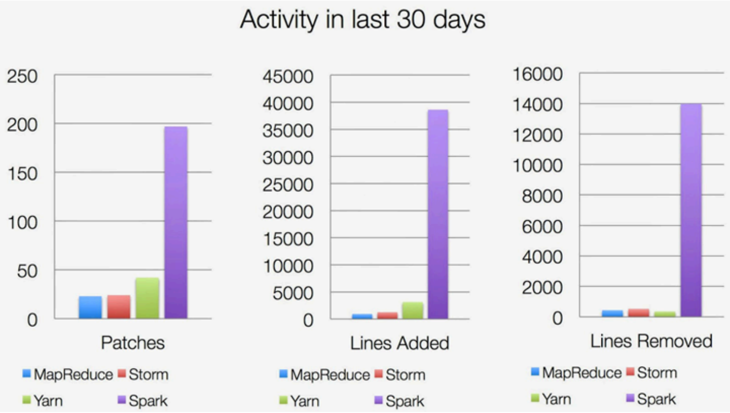
A Growing Number of Applications Are Being Built with Spark

Many more companies want to highlight how they're using Apache Spark in production

by Ben Lorica

One of the trends we're following closely at Strata is the emergence of vertical applications. As components for creating large-scale data infrastructures enter their early stages of maturation, companies are focusing on solving data problems in specific industries rather than building tools from scratch. Virtually all of these components are open source and have contributors across many companies. Organizations are also sharing best practices for building big data applications, through blog posts, white papers, and presentations at conferences like Strata.

These trends are particularly apparent in a set of technologies that originated from UC Berkeley's AMPLab: the number of companies that are using (or plan to use) Spark[1] in production[2] has exploded over the last year. The surge in popularity of the Apache Spark ecosystem stems from the maturation of its individual *open source* components and the growing community of users. The tight integration of *high-performance* tools that address different problems and workloads, coupled with a simple programming interface (in Python, Java, Scala), make Spark one of the most popular projects in big data. The charts below show the amount of active development in Spark:

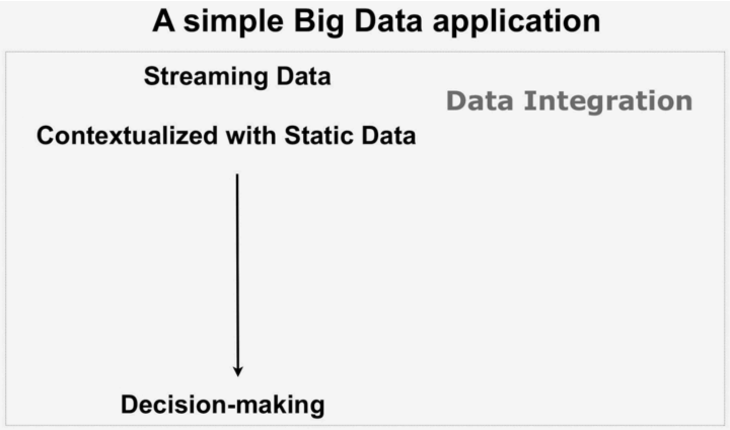


For the second year in a row, I've had the privilege of serving on the program committee for the Spark Summit. I'd like to highlight a few

[1] Full disclosure: I am an advisor to Databricks—a startup commercializing Apache Spark.

[2] I've been on the program committee for all the Spark Summits, and I'm the Content Director of Strata. The number of proposals from companies using Spark has grown considerably over the last year.

areas where Apache Spark is making inroads. I'll focus on proposals[3] from companies building applications on top of Spark.


A simple Big Data application
Streaming Data
Contextualized with Static Data
Data Integration
Decision-making

Real-time processing

This year's conference saw many more proposals from companies that have *deployed* Spark Streaming. The most common usage remains real-time data processing and scoring of machine-learning models. There are also sessions on tools that expand the capability and appeal of Spark Streaming (StreamSQL and a tool for CEP).

Data integration[4] and data processing

At the heart of many big data applications are robust data fusion systems. This year's Spark Summit has several sessions on how companies integrate data sources and make the results usable for services and applications. Nube Technologies is introducing a fuzzy matching engine for handling noisy data, a new library called Geotrellis adds geospatial data processing to Spark, and a new library for parallel, distributed, real-time image processing will be unveiled.

[3] There are also many sessions on core technologies in the the Apache Spark ecosystem.

[4] This is an important topic that's frequently underestimated by engineering managers. For more on this subject, check Jay Kreps' recent Strata webcast on data integration.

Advanced analytics

MLlib, the default machine-learning in the Spark ecosystem, is already used in production deployments. This year's conference includes sessions on tools that enhance MLlib including a distributed implementation of Random Forest, matrix factorization, entity recognition in NLP, and a new library that enables analytics on tabular data (distributed data frames from Adatao).

Beyond these new tools for analytics, there are sessions on how Spark enables large-scale, real-time recommendations (Spotify, Graphflow, Zynga), text mining (IBM, URX), and detection of malicious behavior (FSecure).

Applications

I've written about how Spark is being used in genomics. Spark is also starting to get deployed in applications that generate and consume machine-generated data (including wearables and the Internet of Things). Guavus has recently begun to move parts of their widely deployed application for processing large amounts of event data to Spark.

There are many more interesting applications[5] that will be featured at the upcoming Spark Summit. This community event is fast becoming a showcase of next-generation big data applications. As Spark continues to mature, this shift in focus from tools to case studies and applications will accelerate even more.

New release

June 2014 marked a major milestone for Apache Spark with the release of the 1.0 version. The release notes have a detailed list of enhancements and improvements, but here are a few that should appeal to the data community: a new feature called Spark SQL provides schema-aware data modeling and SQL language support in Spark, MLLib has expanded to include several new algorithms, and there are major updates to both Spark Streaming and GraphX.

[5] Databricks is in the early stages of a new program (Certified on Spark) to certify applications that are compatible with *Apache* Spark.

Why the Data Center Needs an Operating System

It's time for applications—not servers—to rule the data center

by Benjamin Hindman

Developers today are building a new class of applications. These applications no longer fit on a single server, but instead run across a fleet of servers in a data center. Examples include analytics frameworks like Apache Hadoop and Apache Spark, message brokers like Apache Kafka, key-value stores like Apache Cassandra, as well as customer-facing applications such as those run by Twitter and Netflix.

These new applications are more than applications, they are distributed systems. Just as it became commonplace for developers to build multithreaded applications for single machines, it's now becoming commonplace for developers to build distributed systems for data centers.

But it's difficult for developers to build distributed systems, and it's difficult for operators to run distributed systems. Why? Because we expose the wrong level of abstraction to both developers and operators: machines.

Machines are the Wrong Abstraction

Machines are the wrong level of abstraction for building and running distributed applications. Exposing machines as the abstraction to developers unnecessarily complicates the engineering, causing developers to build software constrained by machine-specific characteristics, like IP addresses and local storage. This makes moving and resizing applications difficult if not impossible, forcing maintenance in data centers to be a highly involved and painful procedure.

With machines as the abstraction, operators deploy applications in anticipation of machine loss, usually by taking the easiest and most conservative approach of deploying one application per machine. This almost always means machines go underutilized since we rarely buy our machines (virtual or physical) to exactly fit our applications, or size our applications to exactly fit our machines.

By running only one application per machine, we end up dividing our data center into highly static, highly inflexible partitions of machines, one for each distributed application. We end up with a partition that runs analytics, another that runs the databases, another that runs the web servers, another that runs the message queues, and so on. And the number of partitions is only bound to increase as companies replace monolithic architectures with service-oriented architectures and build more software based on microservices.

What happens when a machine dies in one of these static partitions? Let's hope we over-provisioned sufficiently (wasting money), or can re-provision another machine quickly (wasting effort). What about when the web traffic dips to its daily low? With static partitions we allocate for peak capacity, which means when traffic is at its lowest, all of that excess capacity is wasted. This is why a typical data center runs at only 8-15% efficiency. And don't be fooled just because you're running in the cloud: you're still being charged for the resources your application is not using on each virtual machine (someone is benefiting—it's just your cloud provider, not you).

And finally, with machines as the abstraction, organizations must employ armies of people to manually configure and maintain each individual application on each individual machine. People become the bottleneck for trying to run new applications, even when there are ample resources already provisioned that are not being utilized.

If My Laptop Were a Data Center

Imagine if we ran applications on our laptops the same way we run applications in our data centers. Each time we launched a web browser or text editor, we'd have to specify which CPU to use, which memory modules are addressable, which caches are available, and so on. Thankfully, our laptops have an operating system that abstracts us away from the complexities of manual resource management.

In fact, we have operating systems for our workstations, servers, mainframes, supercomputers, and mobile devices, each optimized for their unique capabilities and form factors.

We've already started treating the data center itself as one massive warehouse-scale computer. Yet, we still don't have an operating system that abstracts and manages the hardware resources in the data center just like an operating system does on our laptops.

It’s Time for the Data Center OS

What would an operating system for the data center look like?

From an operator’s perspective it would span all of the machines in a data center (or cloud) and aggregate them into one giant pool of resources on which applications would be run. You would no longer configure specific machines for specific applications; all applications would be capable of running on any available resources from any machine, even if there are other applications already running on those machines.

From a developer’s perspective, the data center operating system would act as an intermediary between applications and machines, providing common primitives to facilitate and simplify building distributed applications.

The data center operating system would not need to replace Linux or any other host operating systems we use in our data centers today. The data center operating system would provide a software stack on top of the host operating system. Continuing to use the host operating system to provide standard execution environments is critical to immediately supporting existing applications.

The data center operating system would provide functionality for the data center that is analogous to what a host operating system provides on a single machine today: namely, resource management and process isolation. Just like with a host operating system, a data center operating system would enable multiple users to execute multiple applications (made up of multiple processes) concurrently, across a shared collection of resources, with explicit isolation between those applications.

An API for the Data Center

Perhaps the defining characteristic of a data center operating system is that it provides a software interface for building distributed applications. Analogous to the system call interface for a host operating system, the data center operating system API would enable distributed applications to allocate and deallocate resources, launch, monitor, and destroy processes, and more. The API would provide primitives that implement common functionality that all distributed systems need. Thus, developers would no longer need to independently re-implement fundamental distributed systems primitives (and inevitably, independently suffer from the same bugs and performance issues).

Centralizing common functionality within the API primitives would enable developers to build new distributed applications more easily, more safely, and more quickly. This is reminiscent of when virtual memory was added to host operating systems. In fact, one of the virtual memory pioneers wrote that "it was pretty obvious to the designers of operating systems in the early 1960s that automatic storage allocation could significantly simplify programming."

Example Primitives

Two primitives specific to a data center operating system that would immediately simplify building distributed applications are service discovery and coordination. Unlike on a single host where very few applications need to discover other applications running on the same host, discovery is the norm for distributed applications. Likewise, most distributed applications achieve high availability and fault tolerance through some means of coordination and/or consensus, which is notoriously hard to implement correctly and efficiently.

Developers today are forced to pick between existing tools for service discovery and coordination, such as Apache ZooKeeper and CoreOS's etcd. This forces organizations to deploy multiple tools for different applications, significantly increasing operational complexity and maintainability.

Having the data center operating system provide primitives for discovery and coordination not only simplifies development, it also enables application portability. Organizations can change the underlying implementations without rewriting the applications, much like you can choose between different filesystem implementations on a host operating system today.

A New Way to Deploy Applications

With a data center operating system, a software interface replaces the human interface that developers typically interact with when trying to deploy their applications today; rather than a developer asking a person to provision and configure machines to run their applications, developers launch their applications using the data center operating system (e.g., via a CLI or GUI), and the application executes using the data center operating system's API.

This supports a clean separation of concerns between operators and users: operators specify the amount of resources allocatable to each user, and users launch whatever applications they want, using whatever resources are available to them. Because an operator now specifies *how much* of any type of resource is available, but not which *specific* resource, a data center operating system, and the distributed applications running on top, can be more intelligent about which resources to use in order to execute more efficiently and better handle failures. Because most distributed applications have complex scheduling requirements (think Apache Hadoop) and specific needs for failure recovery (think of a database), empowering software to make decisions instead of humans is critical for operating efficiently at data-center scale.

The "Cloud" is Not an Operating System

Why do we need a new operating system? Didn't Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) already solve these problems?

IaaS doesn't solve our problems because it's still focused on machines. It isn't designed with a software interface intended for applications to use in order to execute. IaaS is designed for humans to consume, in order to provision virtual machines that other humans can use to deploy applications; IaaS turns machines into more (virtual) machines, but does not provide any primitives that make it easier for a developer to build distributed applications on top of those machines.

PaaS, on the other hand, abstracts away the machines, but is still designed first and foremost to be consumed by a human. Many PaaS solutions do include numerous tangential services and integrations that make building a distributed application easier, but not in a way that's portable across other PaaS solutions.

Apache Mesos: The Distributed Systems Kernel

Distributed computing is now the norm, not the exception, and we need a data center operating system that delivers a layer of abstraction and a portable API for distributed applications. Not having one is hindering our industry. Developers should be able to build distributed applications without having to reimplement common functionality. Distributed applications built in one organization should be capable of being run in another organization easily.

Existing cloud computing solutions and APIs are not sufficient. Moreover, the data center operating system API must be built, like Linux, in an open and collaborative manner. Proprietary APIs force lock-in, deterring a healthy and innovative ecosystem from growing. It's time we created the POSIX for distributed computing: a portable API for distributed systems running in a data center or on a cloud.

The open source Apache Mesos project, of which I am one of the cocreators and the project chair, is a step in that direction. Apache Mesos aims to be a distributed systems kernel that provides a portable API upon which distributed applications can be built and run.

Many popular distributed systems have already been built directly on top of Mesos, including Apache Spark, Apache Aurora, Airbnb's Chronos, and Mesosphere's Marathon. Other popular distributed systems have been ported to run on top of Mesos, including Apache Hadoop, Apache Storm, and Google's Kubernetes, to list a few.

Chronos is a compelling example of the value of building on top of Mesos. Chronos, a distributed system that provides highly available and fault-tolerant cron, was built on top of Mesos in only a few thousand lines of code and without having to do any explicit socket programming for network communication.

Companies like Twitter and Airbnb are already using Mesos to help run their datacenters, while companies like Google have been using in-house solutions they built almost a decade ago. In fact, just like Google's MapReduce spurred an industry around Apache Hadoop, Google's in-house datacenter solutions have had close ties with the evolution of Mesos.

While not a complete data center operating system, Mesos, along with some of the distributed applications running on top, provide some of the essential building blocks from which a full data center operating system can be built: the kernel (Mesos), a distributed init.d (Marathon/Aurora), cron (Chronos), and more.

Interested in learning more about or contributing to Mesos? Check out mesos.apache.org and follow @ApacheMesos on Twitter. We're a growing community with users at companies like Twitter, Airbnb, Hubspot, OpenTable, eBay/Paypal, Netflix, Groupon, and more.

Design and Social Science

Having established itself as a field with a body of knowledge and established norms and practices, data science is increasingly open to (and recognizing its need for) bringing in other approaches. This chapter kicks off with an overview of how ideation workshops—a mainstay of the field of design—open up new ways of thinking about problems, whether research or business questions. Questions of design are also at the forefront of the exploding area of wearable computing; we are designing not just devices but new experiences, and new ways of structuring personal boundaries.

How Might We...

Human-centered design techniques from an ideation workshop

By Bo Peng and Aaron Wolf of Datascope Analytics

At Datascope Analytics, our ideation workshop combines elements from human-centered design principles to develop innovative and valuable ideas/solutions/strategies for our clients. From our workshop experience, we've developed a few key techniques that have enabled successful communication and collaboration. We complete certain milestones during the workshop: the departure point, the dream view, and curation with gold star voting, among others. These are just a few of the accomplishments that are achieved at various points during the workshop. In addition, we strive to support cultural goals throughout the workshop's duration: creating an environment that spurs creativity and encourages wild ideas, and maintaining a mediator role. These techniques have thus far proven successful in providing innovative and actionable solutions for our clients.

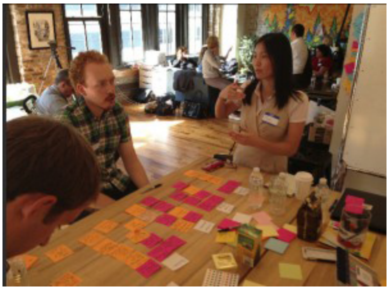


Figure 6-1. Bo Peng at a Datascope Analytics Ideation Workshop in Chicago

Technique #1: The Welcoming Culture

Throughout the workshop, we strive to establish an inclusive, welcoming culture. Brainstorming is an exercise that relies on having high volumes of ideas, so it's vital that everyone be comfortable presenting "wacky" ideas. To maximize participation, we lay out simple rules that help us achieve the ideal workshop environment. For example, everyone should be enthusiastic and encouraging when others express ideas. This is especially critical at the start of brainstorming, when participants are often nervous to speak up for the first time. When appropriate, validating opinions and affirming ideas make people feel welcome and therefore more likely to fully engage in the discussion.

Technique #2: The "Departure Point"

To ensure that every workshop attendee is on the same page, we start the ideation workshop by asking broad, open-ended questions. What are the most important problems they face as a business? What kinds of solutions would make their jobs easier or make them more productive? This spurs discussion amongst our client attendees, which we help guide to reach a consensus. We call this the "departure point" of our workshop—the point at which everyone agrees upon and understands the client's most pressing problems and their underlying issues. We use the ideas and opinions from the departure point to guide our discussions moving forward.

Technique #3: The "Dream View"

After establishing the clients' business problems in the departure point phase, we begin to brainstorm solutions as a group. Our job is not so much to generate great ideas ourselves, rather, it is to encourage others to generate great ideas by helping them think big and by bringing a different perspective to the problem. To that end, we frame the question in order to discover the "dream view"—regarding their business, if they could have anything, what kinds of solutions would they want? In painting their dream view, we focus on what our clients need, not what our clients can achieve given resource constraints. We steer the discussion towards how these dream solutions would look and be used by our clients when solving a tangible problem or deciding new strategic direction for the business and veer away from discussing technical tools or data limitations. Only after we paint the dream view do we cull the the ideas, consider the data and available resources, and synthesize a manageable action plan. Diverge then converge.

Technique #4: Gold Stars and Other Props

Figuring out the dream view and then culling down into viable pieces is no easy task. We are essentially applying structure to seeming chaos; the diverging and converging of so many different thoughts is very difficult. To make this process as fun and effective as possible, we resort to using props throughout the workshop:

One idea, one post-it

> Much of the workshop is designed to maximize the volume and diversity of ideas generated. We have found that post-it sticky notes written in thick permanent marker ink are quite effective. We limit one thought or idea for each post-it, and encourage visual sketches rather than verbose descriptions. Everyone notes their own ideas and posts them on a whiteboard, presenting each idea in a few seconds as they are posted. After the end of an hour-long brainstorming session, there are sometimes hundreds of post-its on several whiteboards!

Voting with gold stars

> This is the first segment of the workshop in which we pass judgment. After we exhaust the possibilities and our idea generation rate naturally halts, we take a break from working together and reflect on the viability, tangibility, and desirability of our ple-

thora of ideas. Each workshop participant is given a few stickers (usually three to five gold stars), walks around the room and "votes" for the ideas they wish to move forward with. At the end of this voting round, up to half a dozen popular ideas clearly emerge.

Drawings/wireframes

We further hone the most popular ideas by exploring possible functionality in more detail. The goal is to create a story that we can share with the rest of group through more refined drawings. If we determine that an interactive online dashboard is the best solution, we make wireframes—skeletal drawings that illustrate the framework and end-user interface. By thinking about how the client will use the tool, we bring to life the seemingly unconnected ideas. All of these drawings are done free-hand, with pencils, markers and crayons, to encourage live feedback from the clients; in our experience, people are much more willing to constructively criticize a rough sketch than a polished one.

Technique #5: On Being a Mediator

We also try to mediate productive, flowing conversation. Tangents naturally come up in discussions and can either contribute or take away from the productivity of the workshop. As a mediator, we must keep the ball in the court! For example, during the dream view discussions, we truncate discussions about resource limitations and direct the conversation flow back to desired, "dream" functionalities. There is a delicate balance of a mediator to encourage wild ideas that think outside of the box while maintaining the ship on the correct course. Experience is key while we have definitely found the bermuda triangle many times in the past!

We at Datascope Analytics are always learning and refining through iteration, and our Ideation workshop is no exception. Through this, we've come to see the value in the general direction of our workshops: of fostering a collaborative environment in which quantity is hailed over quality and in which ideas diverge, and then converging toward some ideas with voting and curation. The departure point, the dream view, and gold star methods have all worked well to achieve this direction. If you would like to learn more, or delve further into the design aspect of our work, we encourage you to attend our upcoming tutorial at Strata Santa Clara.

Wearables and the Immediacy of Communication

Wearables can help bridge the gap between batch and real-time communications

by Matthew Gast

I drown in e-mail, which is a common affliction. With meetings during the day, I need to defer e-mail to breaks between meetings or until the evening, which prevents it from being a real-time communications medium.

Everybody builds a communication "bubble" around themselves, sometimes by design and sometimes by necessity. Robert Reich's memoir *Locked in the Cabinet* describes the process of staffing his office and, ultimately, building that bubble. He resists, but eventually succumbs to the necessity of filtering communications when managing such a large organization.

One of the reasons I'm fascinated by wearable technology is that it is one way of bridging the gap between batch and real-time communications. Wearable technology has smaller screens, and many early products use low-power screen technology that lacks the ability to display vibrant colors. Some may view these qualities as drawbacks, but in return, it is possible to display critical information in an easily viewable—and immediate—way.

Some wearables are also able to alert you by physical feedback. For instance, a wearable device connected to your smartphone can provide vibrating alerts to call your attention to important information when you're in crowded, noisy environments where you might not hear or feel your phone. I recently spent some time wearing a Pebble smart watch, which afforded me a few insights. One is that I was relentlessly teased by my coworkers for having a "crowded wrist."

The experience was worth every jibe and taunt, though. If the Pebble were just a remote display for the phone, it would not be that useful. My phone gives me too much information, and what I needed was a way to get the immediate alerts when it was important to pay attention. The smart watch's configurable rules on what to push to the phone made the experience of interacting with my phone totally different. When the Pebble vibrates, it's worth paying attention: it is a call or text from somebody I've identified as important, or an application from which I've chosen to receive alerts.

One funny way that Pebble offered immediacy was that I found it alerted me to phone calls faster than the phone itself did. My phone is paired to my car, and I use the Bluetooth hands-free profile to talk on the phone when I'm driving. One night, I was driving home and the Pebble vibrated on my wrist. The phone was not yet making noise, so I wasn't sure what to make of the Pebble alert. I looked at my wrist, and saw that there was an incoming phone call. As I read the caller ID off my watch, I heard the phone start to ring. After another half second, the car paused my iPod and alerted me to the incoming call. Surprisingly, the first indication of the phone call came from the Pebble, even though the phone obviously has to handle the call first.

During the time I wore the Pebble, they launched an app store, which was a great move. I had some limited ability to customize the display, and at one point tried to create a pilot-friendly display of groundspeed and altitude to use when I'm flying a glider. I'm not enough of a developer to use the current tools to customize my watch display to any great degree, but that doesn't mean better tools will not become available. As the community grows, it also will be more likely that somebody will have already designed a layout that displays just the information you need.

Making wearables work well and "just fit" into your life potentially requires multiple intersecting disciplines: industrial design, to come up with a product that fits in with a wide variety of clothing (the original Pebble was clearly a geek toy, but the new metal Pebble is suitable for wear with more professional dress, for instance); user interaction design, so that the right information arrives at the small screen at the right time; and maybe even a dash of data processing, so that the filter of what winds up on my wrist can be "trained" over time to recognize what is important to me. The Pebble is a great first step, and I am looking forward to what comes next.

Wearable Intelligence

Establishing protocols to socialize wearable devices

by Glen Martin

The age of ubiquitous computing is accelerating, and it's creating some interesting social turbulence, particularly where wearable hardware is concerned.

Intelligent devices other than phones and screens—smart headsets, glasses, watches, bracelets—are insinuating themselves into our daily lives. The technology for even less intrusive mechanisms, such as jewelry, buttons, and implants, exists and will ultimately find commercial applications.

And as sensor-and-software-augmented devices and wireless connections proliferate through the environment, it will be increasingly difficult to determine who is connected—and how deeply—and how the data each of us generates is disseminated, captured and employed. We're already seeing some early signs of wearable angst: recent confrontations in bars and restaurants between those wearing Google Glass and others worried they were being recorded.

This is nothing new, of course. Many major technological developments experienced their share of turbulent transitions. Ultimately, though, the benefits of wearable computers and a connected environment are likely to prove too seductive to resist. People will participate and tolerate because the upside outweighs the downside.

"It's going to have dramatic effects on the way you live your life—or rather, on how you *choose* to live your life," observes Joe Burton, chief technology officer at Plantronics, a lead manufacturer in intelligent wearable devices.

Burton cites three linked factors driving the wearable revolution:

> First, obviously, is big data. In two years, there will be data equivalent to 72 million Libraries of Congress available online. The second is the ubiquitous, connected network that will make that data instantly available to anyone, anywhere. Finally, we have powerful and rapidly evolving analytics—the means for finding any particular needle you want in the ever-expanding data haystack.

This nexus of wearable intelligent devices and ubiquitous wireless connectivity will greatly amplify our essential powers, says Burton. It will expand our senses and our abilities, and the control we exert over our lives.

“Health is a major application,” he says. “As the Quantified Self trend moves deeper into the mainstream, its advantages will become obvious to people other than technophiles.”

That applies most pertinently to people suffering from chronic conditions.

“Say you’re a 48-year-old man with some cardiac issues,” says Burton. “The connected environment could monitor everything from your blood chemistry to your physical appearance, to your perspiration rate—and match that to a profile of potential cardiac arrest candidates. If you were alerted that you have a 27% chance of suffering a heart attack in the next 48 hours, you’d probably consider that a good thing.”

But in a larger sense, continues Burton, wearables will function as personal concierges, accommodating your specific needs to the connected world. He likens it to the entourage that surrounds the President of the United States wherever he or she goes.

“People are always clustered around the President, looking into screens, talking into headsets, bending over to whisper something urgent in his ear, handing him a paper or a phone with a text message that he needs to read,” observes Burton. “At any given moment, a large number of people are gathering, filtering and organizing the information that he requires. Wearables will essentially perform the same function for all the rest of us. They will be our ‘personalizer’ for the Internet of Things.”

To some, that may seem like information overload, but Burton feels the process will ultimately feel unobtrusive and instinctive; it will merge into our quotidian activities rather than dominate them.

“It’ll manifest in things as basic as returning from work,” he continues. “As you approach your home, sensors you’re wearing will communicate with your house. Your skin temperature will be evaluated; if you’re feeling too warm or cold, the house will adjust the thermostat accordingly. Your identity will be confirmed as you approach, and the door will unlock. Your favorite music will play. If you have medical issues, your stress levels and vital signs will be scanned, and appropriate recommendations will be made. In the end, it will be about streamlining your life, not complicating it.”

That's not to say that all the issues have been resolved on this road to sublime connectedness, of course. As noted earlier, the recent tiffs over Google Glass indicate some basic ground rules have yet to be thrashed out. We'll probably have to expand the concept of "netiquette" to accommodate wearable computing etiquette.

"People have the right to expect they won't be quantified without their consent," Burton says. "That's why there can be objections when someone is wearing Google Glass in a public place. On a personal level, I recognize this. I use Google Glass, but I'm careful where I wear it. I don't want to offend people, or make them feel I'm intruding on them."

Further, society ultimately will have to determine how quantified data will be used; how it is protected; and who, if anyone, gets recompensed.

"There are a couple of lanes to this," says Burton. "First, any specific data that I deliberately generate because I use quantified sensors on myself and at home, and that can be linked to me, should be mine. I must have reasonable security, and if I choose to share it, I have to fully understand what I'm giving up and what services or compensation I can receive in return."

But we will also generate anonymous quantified data that can enhance the public good, continues Burton—and that's a different matter. Truly anonymous data—say, metadata that can be used to determine health risks, or track atmospheric pollution plumes, or predict traffic patterns—have great potential for improving all our lives with virtually no negative impact to the people generating the information.

"We should be willing to share that," says Burton. "It benefits us all, and it's simply part of being a good citizen."

Data generation and utilization concerns are also apt to drive the configuration of wearable devices, says Burton. It's already possible to design ubiquitous computing systems that fade deep into the background. But do we want the devices to disappear from sight entirely? Probably not.

"Certainly, if you're talking about quantifying your own physical data, you'd be agreeable to small devices like rings or even skin [decals] or implants," says Burton. "But for other devices—those that gather data from your environment or other people—not so much.

That doesn’t mean you necessarily want to wear headsets or clunky bracelets. We may see a kind of socially acceptable standard evolving for the size of wearable devices: smaller than headsets, but bigger than jewelry. They’ll still be unobtrusive, but they’ll be large enough to signal their function.”

Building a Data Culture

In an environment where new capacities and tools are introduced almost daily, the limiting challenge in an organization is often harnessing these powers to drive decisions and action. While there are many good ways to structure your data personnel and their relationship to the rest of the organization, mutual understanding of needs (the organization's) and capacities (the data team's) are vital. In this chapter, Chris Diehl takes us into the life-and-death decisions that the U.S. military was confronting in the Iraq war; data analysis needed to be reframed, and the structures of intelligence gathering and action on the basis of that intelligence rebuilt, in order to save lives. Few other problems present this kind of urgency, but the questions that any organization needs to ask itself are the same. Mike Loukides then steps back and offers a reminder: not only do you have to ask the right questions, you have to make clear what the questions are and why they are important in order to make data analysis actionable.

Understanding the Now: The Role of Data in Adaptive Organizations

Focusing attention on the present lets organizations pursue existing opportunities as opposed to projected ones

by Chris Diehl of The Data Guild

Slow and Unaware

It was 2005. The war in Iraq was raging. Many of us in the national security R&D community were developing responses to the deadliest threat facing U.S. soldiers: the improvised explosive device (IED). From the perspective of the U.S. military, the unthinkable was happening each and every day. The world's most technologically advanced military was being dealt significant blows by insurgents making crude weapons from limited resources. How was this even possible?

The war exposed the limits of our unwavering faith in technology. We depended heavily on technology to provide us the advantage in an environment we did not understand. When that failed, we were slow to learn. Meanwhile the losses continued. We were being disrupted by a patient, persistent organization that rapidly experimented and adapted to conditions on the ground.

To regain the advantage, we needed to start by asking different questions. We needed to shift our focus from the devices that were destroying U.S. armored vehicles to the people responsible for building and deploying the weapons. This motivated new approaches to collect data that could expose elements of the insurgent network.

New organizations and modes of operation were also required to act swiftly when discoveries were made. By integrating intelligence and special operations capabilities into a single organization with crisp objectives and responsive leadership, the U.S. dramatically accelerated its ability to disrupt insurgent operations. Rapid orientation and action were key in this dynamic environment where opportunities persisted for an often unknown and very limited period of time.

This story holds important and under appreciated lessons that apply to the challenges numerous organizations face today. The ability to collect, store, and process large volumes of data doesn't confer advantage by default. It's still common to fixate on the wrong questions and fail to recover quickly when mistakes are made. To accelerate organizational learning with data, we need to think carefully about our objectives and have realistic expectations about what insights we can derive from measurement and analysis.

Embracing Uncertainty

In recent years, decision makers have embraced a number of simplistic misconceptions. One of particular concern is the idea that our ability to predict reliably improves with the volume of available data. Unfortunately reality is more complex.

One of the key drivers of prediction performance is the stability of the environment. When environmental conditions change, our ability to predict often degrades. No amount of historical data will inform us about the duration of a particular pattern or the nature of the change to follow.

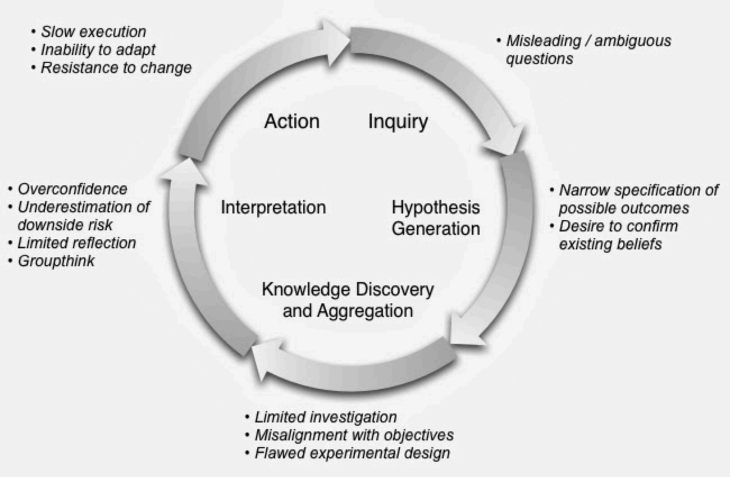
Our globalized world relies on complex, interconnected systems that produce enormous volumes of data; yet network effects and cascading failures routinely surprise us. In many ways, we know more than ever about the present; meanwhile the future remains stubbornly uncertain.

Data-driven prediction is viewed by some as a potential antidote to the risk associated with delay in action. This is a dangerous belief in complex environments. Overconfidence coupled with delay significantly magnifies the cost of prediction errors when they occur.

From Prediction to Adaptation

To combat this, a shift in mindset is required. We need to shift from predicting the future to understanding the now. By focusing our attention on the present, we uncover and pursue existing opportunities as opposed to projected ones that may never come to pass. By accelerating our pace of response, we increase our potential to benefit from surprises that will surely come. At the same time, we mitigate the cost of our mistakes.

Difficulties in the organizational learning process can take many forms. The following diagram highlights problems that commonly arise at different stages of the process:


• Slow execution
• Inability to adapt
• Resistance to change
• Misleading / ambiguous questions
Action
Inquiry
• Overconfidence
• Underestimation of downside risk
• Limited reflection
• Groupthink
Interpretation
Hypothesis Generation
• Narrow specification of possible outcomes
• Desire to confirm existing beliefs
Knowledge Discovery and Aggregation
• Limited investigation
• Misalignment with objectives
• Flawed experimental design

In many respects, the challenges we face are struggles with preconceptions at both the individual and group level. Adaptation requires an openness to alternatives and a rejection of the temptation to simply confirm existing beliefs. Leadership is absolutely key to foster a culture where curiosity and experimentation are core values.

In an adaptive organization, measurement and analysis can be valuable tools for understanding the present environment and evaluating the effectiveness of our actions. Advances in Internet and mobile technologies have dramatically expanded the scope and rate at which certain types of information can be collected. With clearly defined objectives, these capabilities can be leveraged to uncover opportunities much more rapidly.

Once a course of action has been selected and implemented, adaptive organizations also reflect on both the derived benefit and the efficiency of execution. Measurement and analysis can illuminate the resulting changes in the environment and the level of time and effort required to achieve that outcome. This can serve as the basis for more thoughtful discussion of ways to accelerate the organization's response to changing conditions.

A Holistic View

Too often when lauding the potential of data-driven decision-making, the technology sector focuses solely on the data analytic tools they believe are central to supporting their envisioned future. All the while, more fundamental organizational issues determine the ultimate impact of data in the learning process. As the IED threat in Iraq made clear, an unwillingness to adapt coupled with sluggish action can have dramatic consequences in a dynamic environment. Only a dogged focus on present opportunities coupled with efficiency of action will mitigate the risks of a persistently uncertain future. Data can be a powerful resource for accelerating the learning process. Yet organizational culture and leadership remain central determinants of the organization's ability to effectively leverage its potential.

Many thanks to Chrys Wu, Mark Huberty, and Beau Cronin for helpful discussions.

The Backlash against Big Data, Continued

Ignore the Hype. Learn to be a data skeptic.

by Mike Loukides

Yawn. Yet another article trashing "big data," this time an op-ed in the Times. This one is better than most, and ends with the truism that data isn't a silver bullet. It certainly isn't.

I'll spare you all the links (most of which are much less insightful than the Times piece), but the backlash against "big data" is clearly in full swing. I wrote about this more than a year ago, in my piece on data skepticism: data is heading into the trough of a hype curve, driven by overly aggressive marketing, promises that can't be kept, and spurious claims that, if you have enough data, correlation is as good as causation. It isn't; it never was; it never will be. The paradox of data is that the more data you have, the more spurious correlations will show up. Good data scientists understand that. Poor ones don't.

It's very easy to say that "big data is dead" while you're using Google Maps to navigate downtown Boston. It's easy to say that "big data is dead" while Google Now or Siri is telling you that you need to leave 20 minutes early for an appointment because of traffic. And it's easy to say that “big data is dead” while you’re using Google, or Bing, or DuckDuckGo to find material to help you write an article claiming that big data is dead.

Big data isn’t dead, though I only use the word “big” under duress. It’s just data. There’s more of it around than there used to be; we have better tools to generate, capture, and store it. As I argued in the beginning of 2013, the mere existence of data will drive the exploration and analysis of data. There’s no reason to believe this will stop.

That said, let’s look at one particular point from the Times op-ed: successful data analysis depends critically on asking the right question. It’s not so much a matter of “garbage in, garbage out” as it is “ask the wrong question, you get the wrong answer.” And here, the author of the Times piece is at least as uncritical as the data scientists he’s criticizing. He criticizes Steven Skiena and Charles Ward, authors of Who is Bigger, along with MIT’s Pantheon project, for the claim that Francis Scott Key was the 19th most important poet in history, and Jane Austin was only the 78th most important writer, and George Eliot the 380th.

Of course, this hinges on the meaning of “important.” If “important” means “central to the musical or literary canon,” then yes, the data-driven results are nonsense. But I wouldn’t expect data analysis to give me the same results I could get by talking to musicologists or literature professors. If by important, we mean that the works somehow drove historical events, I would expect the author of “The Star Spangled Banner” (to say nothing of the authors of “The Marseillaise”) to outrank Keats. People don’t fight wars citing Keats’ Ode on a Grecian Urn.

The Pantheon project doesn’t use the word “important”; it measures global historical popularity, which is something quite different. And their result just isn’t very surprising. It is easy to forget how many authors there are; coming in 78th is not a bad showing when you’re competing with Homer, Shakespeare, and Dante. I am certainly not in a position to debate whether Austen is more or less popular than the Japanese 17th century author Basho (52) or, for that matter, Nostradamus (20).

What do we mean by importance? What do we mean by influence? What do we mean by popularity? These are the sorts of questions you have to ask before doing any data analysis. I haven’t read *Who is Bigger*, but the Pantheon site does an excellent job of discussing its methodology, biases and limitations. And it provides an excellent foundation for a more important, nuanced discussion of popularity, influence, and importance.

There is a lot of hype about "big data," and much of it is ridiculous. Ignore the hype. Learn to be a data skeptic. That doesn't mean becoming skeptical about the value of data; it means asking the hard questions that anyone claiming to be a data scientist should ask. Think carefully about the questions you're asking, the data you have to work with, and the results that you're getting. And learn that data is about enabling intelligent discussions, not about turning a crank and having the right answer pop out.

Data is data. It was valuable 50 years ago, when IBM released the first model 360. It's more valuable today.

The Perils of Big Data

This chapter leads off with a relatively sunny view of data and personal freedom: Jonas Luster's account of his satisfying relationship with Google. The clouds on the horizon come in with Tim O'Reilly's question: "When does a company's knowledge about me cross over into creepiness?" The government's approaches to data privacy are then recounted, and finally two practitioners in the area of data, a lawyer and an executive cut to the chase: "Big Data is about much more than just correlating database tables and creating pattern recognition algorithms. It's about money and power."

One Man Willingly Gave Google His Data. See What Happened Next.

Google requires quid for its quo, but it offers something many don't: user data access

by Jonas Luster

Despite some misgivings about the company's product course and service permanence (I was an early and fanatical user of Google Wave), my relationship with Google is one of mutual symbiosis. Its "better mousetrap" approach to products and services, the width and breadth of online, mobile, and behind-the-scenes offerings saves me countless hours every week in exchange for a slice of my private life, laid bare before its algorithms and analyzed for marketing purposes.

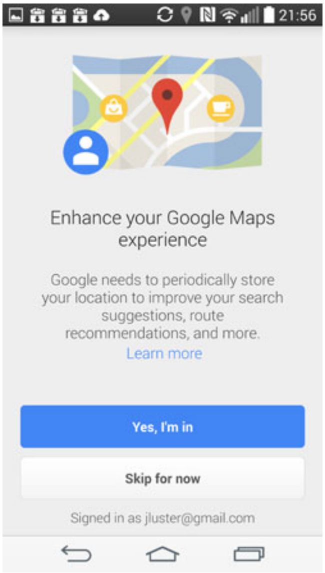
I am writing this on a Chromebook by a lake, using Google Docs and images in Google Drive. I found my way here, through the thick underbrush along a long since forgotten former fishmonger's trail, on Google Maps after Google Now offered me a glimpse of the place as one of the recommended local attractions.

Figure 8-1. The lake I found via Google Maps and a recommendation from Google Now.

Admittedly, having my documents, my photos, my to-do lists, contacts, and much more on Google, depending on it as a research tool and mail client, map provider and domain host, is scary. And as much as I understand my dependence on Google to carry the potential for problems, the fact remains that none of those dependencies, not one shred of data, and certainly not one iota of my private life, is known to the company without my explicit, active, consent.

Just a few weeks ago saw me, once again, doing the new gadget dance. After carefully opening the box and taking in that new phone smell, I went through the onboarding for three phones—Windows, iOS, and Android—for a project. Letting the fingers do the dance they so well know by now, I nevertheless stop every time to read the consent screens offered to me by Apple, Google, and others. "Would you like to receive an email every day reminding you to pay us more money?"—No. "Would you like to sign up for an amazing newsletter containing no news but lots of letters?"—No. "Google needs to periodically store your location to improve your search suggestions, route recommendations, and more"—Yes.

"You would never believe what Google secretly knows about you," says the headline in my Facebook feed. Six of my friends have so far re-shared it, each of whom expresses their dismay about yet another breach of privacy, inevitably containing sentence fragments such as "in a post-Snowden world" and calling Google's storage and visualization of a user's location data "creepy."



Enhance your Google Maps experience

Google needs to periodically store your location to improve your search suggestions, route recommendations, and more.

Signed in as jluster@gmail.com

This is where the narrative, one about privacy and underhanded dealings, splits from reality. Reality comes with consent screens like the one pictured to the right and a "Learn more" link. In reality the "creepy" part of this event isn't Google's visualization of consensually shared data on its Location History page, it's the fact that the men and women whom I hold in high esteem as tech pundits and bloggers, apparently click consent screens without reading them. Given the publicity of Latitude on release and every subsequent rebranding and reshaping, and an average of 18 months between device onboarding for the average geek, it takes quite a willful ignorance to not be aware of this feature.

And a feature it is. For me and Google both. Google gets to know where I have been, allowing it to build the better mousetrap it needs to keep me entertained, engaged, and receptive to advertisement. Apparently this approach works: at $16 billion for the second quarter of 2014, Google can't complain about lack of sales.

I get tools and data for my own use as well. Unlike Facebook, OKCupid, Path, and others, Google even gives me a choice and access to my own data at any time. I can start or stop its collection, delete it in its entirety, and export it at any time.

The issue here isn't with Google at all and, at the same time, one of Google's making. By becoming ubiquitous and hard to avoid, offering powerful yet easy-to-use tools, Google becomes to many a proof-positive application of Clarke's Third Law: indistinguishable from magic.

And, like magic, lifting the curtain isn't something many entertain. Clicking the "read more" link, finding links to Google's Dashboard, Location History, and Takeout seems to have been a move so foreign even tech pundits never attempted it. Anyone finding their data on Google's Location History page once consented to the terms of that transaction: Google gets data, user gets better search, better location services, and—in the form of that Location History Page—a fancy visualization and exportable data to boot.

Can Google be faulted for this? Yes, a little bit. Onboarding is one of those things we do more or less on auto pilot. Users assume that declining a consent screen will deprive them of features on their mobile devices. In the case of Google's Location History that's even true, free magic in exchange for a user's life, laid bare before the dissecting data scalpels of the company's algorithm factory.

There is no such thing as a free lunch. We are Google's product, a packaged and well-received $16 billion cluster of humans, sharing our lives with a search engine. Strike Google, replace value and function, and the same could be said for any free service on the Internet, from magazine to search engine, social network to picture-sharing site. In all those cases, however, only Google offers as comprehensive a toolbox for those willing to sign the deal, data for utility.

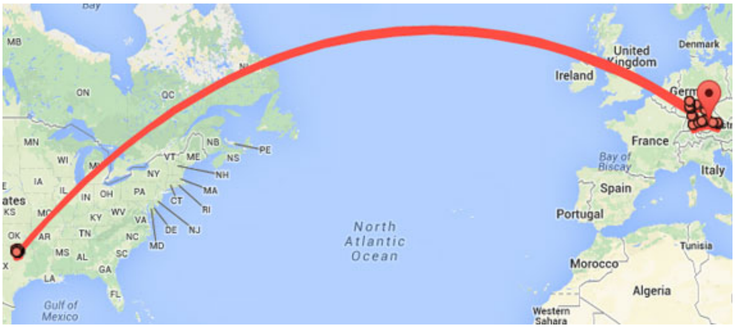
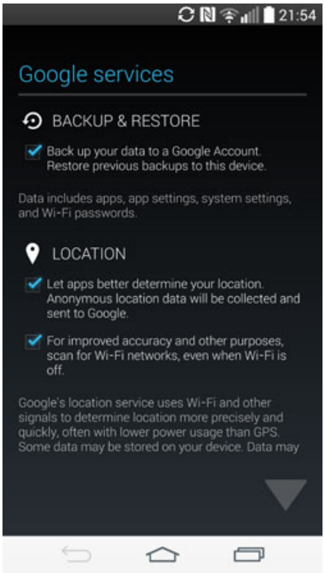


Figure 8-2. My 2010 trip to Germany convinced me to move to the country. In 2013, I replayed the trip to revisit the places that led to this decision.

This makes Google inherently more attackable. The Location History visualizer provides exactly the kind of visceral link ("check out what Google is doing to your phone, you won't believe what I found out they know about you") to show the vastness of the company's data storage; that's tangible, rather than Facebook's blanket "we never delete anything." Hint to the next scare headline writer: Google doesn't just do this for Location, either. Search history traces, if enabled and not deleted, back to the first search our logged-in selves performed on the site (my first recorded, incidentally, was a search for PHPs implode function on April 21, 2005). YouTube viewing history? My first video was (I am now properly ashamed) a funny cat one.

Google doesn't forget. Unless asked to do so, which is more than can be expected from many of the other services out there. That dashboard link, so prominent on every help page linked from each of Google's consent screens, contains tools to pause, resume, delete, or download our history.



Google services

BACKUP & RESTORE

☑ Back up your data to a Google Account. Restore previous backups to this device.

Data includes apps, app settings, system settings, and Wi-Fi passwords.

LOCATION

☑ Let apps better determine your location. Anonymous location data will be collected and sent to Google.

☑ For improved accuracy and other purposes, scan for Wi-Fi networks, even when Wi-Fi is off.

Google's location service uses Wi-Fi and other signals to determine location more precisely and quickly, often with lower power usage than GPS. Some data may be stored on your device. Data may

Google's quo, the collection of data about me to market to me and show me "relevant" ads on Gmail, YouTube, and Search, as well as the ever-growing number of sites running AdWords, begets my quid —better search, better recommendations for more funny cat videos, and an understanding that my search for "explode" might concern PHP, not beached whales.

If there is a click bait headline that should make it onto Facebook, it's not this fake outrage about consensual data collection. It should be one about consent screens. Or, better, one about an amazing life-saver you have to try out that takes your location history, runs it through a KML to GPX converter, and uses it to reverse geotag all those pictures your $2,000 DSLR didn't because the $600 attachment GPS once again failed. Here's how to do it:

1. Open Google Location History and find the download link for the day in question. To add more data points click the "Show all points" link before downloading the KML file.
2. Convert the file to GPX. Most reverse geocoders can not read KML, which means we'll have to convert this file into GPX. Luckily there are a number of solutions, the easiest by far is using a GPX2KML.com. Change the encoding direction in the

dropdown, upload your KML file, download the converted GPX.

3. Use a Geocoding application. Jeffrey Friedl's "Geocode" plugin for Lightroom 5 (and possibly 4) does a good job at this, as does Lightroom 5's built in mechanism. Personally I use Geotag, a free (open source) Java application which also allows me to correct false locations due to jitter before coding my photos.
4. There is no step 4. Enjoy your freshly geocoded images courtesy of Google's quo for your quid.

The Creep Factor

How to think about big data and privacy

by Tim O'Reilly

There was a great passage in Alexis Madrigal's recent interview with Gibu Thomas, who runs innovation at Walmart:

> Our philosophy is pretty simple: When we use data, be transparent to the customers so that they can know what's going on. There's a clear opt-out mechanism. And, more important, the value equation has to be there. If we save them money or remind them of something they might need, no one says, "Wait, how did you get that data?" or "Why are you using that data?" They say, "Thank you!" I think we all know where the creep factor comes in, intuitively. Do unto others as you want to be done to you, right?

This notion of "the creep factor" seems fairly central as we think about the future of privacy regulation. When companies use our data for our benefit, we know it and we are grateful for it. We happily give up our location data to Google so they can give us directions, or to Yelp or Foursquare so they can help us find the best place to eat nearby. We don't even mind when they keep that data if it helps them make better recommendations in future. Sure, Google, I'd love it if you can do a better job predicting how long it will take me to get to work at rush hour! And yes, I don't mind that you are using my search and browsing habits to give me better search results. In fact, I'd complain if someone took away that data and I suddenly found that my search results just weren't as good as they used to be!

But we also know when companies use our data against us, or sell it on to people who do not have our best interests in mind.

When credit was denied not because of your ability to pay but because of where you lived or your racial identity, that was called "redlining," so called because of the practice of drawing a red line on the map to demarcate geographies where loans or insurance would be denied or made more costly. Well, there's a new kind of redlining in the 21st century. *The Atlantic* calls it data redlining:

> When a consumer applies for automobile or homeowner insurance or a credit card, companies will be able to make a pretty good guess as to the type of risk pool they should assign the consumer to. The higher-risk consumers will never be informed about or offered the best deals. Their choices will be limited.
>
> State Farm is currently offering a discount to customers through a program called Drive Safe & Save. The insurer offers discounts to customers who use services such as Ford's Sync or General Motors' OnStar, which, among other things, read your odometer remotely so that customers no longer have to fuss with tracking how many miles they drive to earn insurer discounts. How convenient!
>
> State Farm makes it seem that it's only your mileage that matters but imagine the potential for the company once it has remote access to your car. It will know how fast you drive on the freeway even if you don't get a ticket. It will know when and where you drive. What if you drive on routes where there are frequent accidents? Or what if you park your car in high-crime areas?

In some ways, the worst case scenario in the last paragraph above is tinfoil hat stuff. There is no indication that State Farm Insurance is actually doing those things, but we can see from that example where the boundaries of fair use and analysis might lie. It seems to me that insurance companies are quite within their rights to offer lower rates to people who agree to drive responsibly, and to verify the consumer's claims of how many miles they drive annually, but if my insurance rates suddenly spike because of data about formerly private legal behavior, like the risk profile of where I work or drive for personal reasons, I have reason to feel that my data is being used unfairly against me.

Similarly, if I don't have equal access to the best prices on an online site, because the site has determined that I have either the capacity or willingness to pay more, my data is being used unfairly against me.

The right way to deal with data redlining is not to prohibit the collection of data, as so many misguided privacy advocates seem to urge, but rather, to prohibit its misuse once companies have that data. As David Brin, author of the prescient 1998 book on privacy, *The Transparent Society*, noted in a conversation with me last night, "It is intrinsically impossible to know if someone *does not* have information about you. It is much easier to tell if they *do* something to you."

Furthermore, because data is so useful in personalizing services for our benefit, any attempt to prohibit its collection will quickly be outrun by consumer preference, much as the Germans simply routed around France's famed Maginot Line at the outset of World War II. For example, we are often asked today by apps on our phone if it's OK to use our location. Most of the time, we just say "yes," because if we don't, the app just won't work. Being asked is an important step, but how many of us actually understand what is being done with the data that we have agreed to surrender?

The right way to deal with data redlining is to think about the possible harms to the people whose data is being collected, and primarily to regulate those harms, rather than the collection of the data itself, which can also be put to powerful use for those same people's benefit. When people were denied health coverage because of preexisting conditions, that was their data being used against them; this is now restricted by the Affordable Care Act. By contrast, the privacy rules in HIPAA, the 1996 Health Information Portability and Accountability Act, which seek to set overly strong safeguards around the privacy of data, rather than its use, have had a chilling effect on many kinds of medical research, as well as patients' access to their very own data!

Another approach is shown by legal regimes such as that controlling insider trading: once you have certain data, you are subject to new rules, rules that may actually encourage you to avoid gathering certain kinds of data. If you have material nonpublic data obtained from insiders, you can't trade on that knowledge, while knowledge gained by public means is fair game.

I know there are many difficult corner cases to think through. But the notion of whether data is being used for the benefit of the customer who provided it (either explicitly, or implicitly through his or her behavior), or is being used against the customer's interests by the party that collected it, provides a pretty good test of whether or not we should consider that collecting party to be "a creep."

Big Data and Privacy: An Uneasy Face-Off for Government to Face

MIT workshop kicks off Obama campaign on privacy

by Andy Oram

Thrust into controversy by Edward Snowden's first revelations last year, President Obama belatedly welcomed a "conversation" about privacy. As cynical as you may feel about US spying, that conversation with the federal government has now begun. In particular, the first of three public workshops took place Monday at MIT.

Given the locale, a focus on the technical aspects of privacy was appropriate for this discussion. Speakers cheered about the value of data (invoking the "big data" buzzword often), delineated the tradeoffs between accumulating useful data and preserving privacy, and introduced technologies that could analyze encrypted data without revealing facts about individuals. Two more workshops will be held in other cities, one focusing on ethics and the other on law.

A Narrow Horizon for Privacy

Having a foot in the hacker community and hearing news all the time about new technical assaults on individual autonomy, I found the circumscribed scope of the conference disappointing. The consensus on stage was that the collection of personal information was toothpaste out of the tube, and that all we could do in response was promote oral hygiene. Much of the discussion accepted the conventional view that deriving value from data has to play tug of rope with privacy protection. But some speakers fought that with the hope that technology could produce a happy marriage between the rivals of data analysis and personal data protection.

No one recognized that people might manage their own data and share it at their discretion, an ideal pursued by the Vendor Relationship Management movement and many health care reformers. As an audience member pointed out, no one on stage addressed technologies that prevent the collection of personal data, such as TOR onion routing (which was sponsored by the US Navy).

Although speakers recognized that data analysis could disadvantage individuals, either through errors or through efforts to control us, they barely touched on the effects of analysis on groups.

Finally, while the Internet of Things was mentioned in passing and the difficulty of preserving privacy in an age of social networking was mentioned, speakers did not emphasize the explosion of information that will flood the Internet over the upcoming few years. This changes the context for personal data, both in its power to improve life and its power to hurt us.

One panelist warned that the data being collected about us increasingly doesn't come directly from us. I think that's not yet true, but soon it may be. The Boston Globe just reported that a vast network of vehicle surveillance is run by private industry, unfettered by the Fourth Amendment or discrimination laws (and providing police with their data). If people can be identified by the way they walk, privacy may well become an obsolete notion. But I'm not ready to give up yet on data collection.

In any case, I felt honored to hear and interact with the impressive roster of experts and the well-informed audience members who showed up on Monday. Just seeing Carol Rose of the Massachusetts ACLU sit next to John DeLong of the NSA would be worth a trip downtown. A full house was expected, but a winter storm kept many potential attendees stuck in Washington, DC or other points south of Boston.

Questions the Government is Asking Itself, and Us

John Podesta, a key adviser to the Clinton and Obama administrations, addressed us by phone after the winter storm grounded his flight. He referred to the major speech delivered by President Obama on January 17, 2014, and said that Podesta was leading a working group formed afterward to promote an "open, interoperable, secure, and reliable Internet."

It would be simplistic, however, to attribute Administration interest in privacy to the flak emerging from the Snowden revelations. The government has been trying to cajole industries to upgrade security for years, and launched a cybersecurity plan at the same time as Podesta's group. Federal agencies have also been concerned for some time with promoting more online collaboration and protecting the privacy of participants, notably in the National Strategy for Trusted Identities in Cyberspace (NSTIC) run by the National Institute of Standards and Technology (NIST). (Readers interested in the national approach to identity can find Alexander Howard's analysis on Radar.)

Yes, I know, these were the same folks who passed NSA mischief on to standards committees, seriously weakening some encryption mechanisms. These incidents can remind us that the government is a large institution pursuing different and sometimes conflicting goals. We don't have to withdraw on them on that account and stop pressing our values and issues.

The relationship between privacy and identity may not be immediately clear, but a serious look at one must involve the other. This understanding underscores a series I wrote on identity.

Threats to our autonomy don't end with government snooping. Industries want to know our buying habits and insurers want to know our hazards. MIT professor Sam Madden said that data from the sensors on cell phones can reveal when automobile drivers make dangerous maneuvers. He also said that the riskiest group of drivers (young males) reduce risky maneuvers up to 78% if they know they're being monitored. How do you feel about this? Are you viscerally repelled by such move-by-move snooping? What if your own insurance costs went down and there were fewer fatalities on the highways?

But there is no bright line dividing government from business. Many commenters complained that large Internet businesses shared user data they had collected with the NSA. I have pointed out that the concentration of Internet infrastructure made government surveillance possible.

Revelations that the NSA collected data related to international trade, even though there's no current evidence it is affecting negotiations, makes one wonder whether government spies have cited terrorism as an excuse for pursuing other goals of interest to businesses, particularly when we were tapping the phone calls of leaders in allies such as Germany and Brazil.

Podesta said it might be time to revisit the Fair Information Practices that have guided laws in both the US and many other countries for decades. (The Electronic Privacy Information Center has a nice summary of these principles.)

Podesta also identified a major challenge to our current legal understanding of privacy: the shift from predicated searching to non-predicated or pattern searching. This jargon can be understood as follows: searching for a predicate can be a simple database query to verify a relationship you expect to find, such as whether people who reserve hotel rooms also reserve rental cars. A non-predicated search would turn up totally unanticipated relationships, such as the famous incident where a retailer revealed a customer's pregnancy.

Podesta asked us to consider what's different about big data, what business models are based on big data, what uses there are for big data, and whether we need research on privacy protection during analytics. Finally, he promised a report about three months from now about law enforcement.

Later in the day, US Secretary of Commerce Penny Pritzker offered some further questions: What principles of trust do businesses have to adopt? How can privacy in data be improved? How can we be more accountable and transparent? How can consumers understand what they are sharing and with whom? How can government and business reduce the unanticipated harm caused by big data?

Incentives and Temptations

The morning panel trumpeted the value of data analysis, while acknowledging privacy concerns. Panelists came from medicine, genetic research, the field of transportation, and education. Their excitement over the value of data was so infectious that Shafi Goldwasser of the MIT Computer Science and Artificial Intelligence Laboratory later joked that it made her want to say, "Take my data!"

I think an agenda lay behind the choice of a panel dangling before us an appealing future when we can avoid cruising for parking spots, can make better use of college courses, and can even cure disease through data sharing. In contrast, the people who snoop on social networking sites in order to withdraw insurance coverage from people were not on the panel, and would have had a harder time justifying their use of data. Their presence would highlight the deceptive enticements of data snooping. Big data offers amazing possibilities in the aggregate. Statistics can establish relationships among large populations that unveil useful advice to individuals. But judging each individual by principles established through data analysis is pure prejudice. It leads to such abuses as labeling a student as dissolute because he posts a picture of himself at a party, or withdrawing disability insurance from someone who dares to boast of his capabilities on a social network.

Having Our Cake

Can technology save us from a world where our most intimate secrets are laid at the feet of large businesses? A panel on privacy enhancing techniques suggested it may.

Data analysis without personal revelations is the goal; the core techniques behind it are algorithms that compute useful results from encrypted data. Normally, encrypted data is totally random in principle. Traditionally, it would violate the point of encryption if any information at all could be derived from such data. But the new technologies relax this absolute randomness to allow someone to search for values, compute a sum, or do more complex calculations on encrypted values.

Goldwasser characterized this goal as extracting data without seeing it. For instance, suppose we could determine whether any faces in a surveillance photo match suspects in a database without identifying innocent people in the photo? What if we could uncover evidence of financial turmoil from the portfolios of stockholders without knowing what is held by each stockholder?

Nickolai Zeldovich introduced his CryptDB research, which is used by Google for encrypted queries in BigQuery. CryptDB ensures that any value will be represented by the same encrypted value everywhere it appears in a field, and can also support some aggregate functions. This means you can request the sum of values in a field and get the right answer without having access to any individual values. Different layers of protection can be chosen, each trading off functionality for security to a different degree.

MIT professor Vinod Vaikuntanathan introduced homomorphic encryption, which produces an encrypted result from encrypted data, allowing the user to get the result without seeing any of the input data. This is one of the few cutting-edge ideas introduced at the workshop. Although homomorphic encryption was suggested in 1979, no one could figure out how to make it work till 2009, and viable implementations such as HELib and HCrypt emerged only recently.

The white horse that most speakers wanted to ride is "differential privacy," an unintuitive term that comes from a formal definition of privacy protection: any result returned from a query would be substantially the same whether or not you were represented by a record in that data. When differential privacy is in place, nobody can re-identify your record or even know whether you exist in the database, no matter how much prior knowledge they have about you. A related term is "synthetic data sets," which refers to the practice of offering data sets that are scrambled and muddied by random noise. These data sets are carefully designed so that queries can produce the right answer (for instance, "how many members are male and smoke but don't have cancer?"), but no row of data corresponds to a real person.

Cynthia Dwork, a distinguished scientist at Microsoft Research and one of the innovators in differential privacy, presented an overview that was fleshed out by Harvard professor Salil Vadhan. He pointed out that such databases make it unnecessary for a privacy expert to approve each release of data, because even a user with special knowledge of a person can't re-identify him.

These secure database queries offer another level of protection: checking the exact queries that people run. Vaikuntanathan indicated that homomorphic encryption would be complemented by a functional certification service, which is a kind of mediator that accepts queries from users. The server would check a certificate to ensure the user has the right to issue that particular query before carrying it out on the database.

The ongoing threat to these technologies is the possibility of chipping away at privacy by submitting many queries, possibly on multiple data sets, that could cumulatively isolate the information on a particular person. Other challenges include:

- They depend on data sets big enough to hide individual differences. The bigger the data, the less noise has to be introduced to hide differences. In contrast, small data sets can't be protected well.
- They don't protect the rights of a whole group.
- Because they hide individuals, they can't be used by law enforcement or similar users to target those individuals.

The use of these techniques will also require changes to laws and regulations that make assumptions based on current encryption methods.

Technology lawyer Daniel Weitzner wrapped up the panel on technologies by describing technologies that promote information accountability: determining through computational monitoring how data is used and whether a use of data complies with laws and regulations.

There are several steps to information accountability:

1. First, a law or regulation has to be represented by a "policy language" that a program can interpret.
2. The program has to run over logs of data accesses and check each one against the policy language.
3. Finally, the program must present results with messages a user can understand. Weitzner pointed out that most users want to do the right thing and want to comply with the law, so the message must help them do that.

Challenges include making a policy language sufficiently expressive to represent the law without become too complex for calculations. The language must also allow incompleteness and inconsistency, because laws don't always provide complete answers.

The last panel of the day considered some amusing and thought-provoking hypothetical cases in data mining. Several panelists dismissed the possibility of restricting data collection but called for more transparency in its use. We should know what data is being collected and who is getting it. One panelist mentioned Deborah Estrin, who calls for companies to give us access to "data about me." Discarding data after a fixed period of time can also protect us, and is particularly appealing because old data is often no use in new environments.

Weitzner held out hope on the legal front. He suggested that when President Obama announced a review of the much-criticized Section 215 of the Patriot Act, he was issuing a subtle message that the fourth amendment would get more consideration. Rose said that revelations about the power of metadata prove that it's time to strengthen legal protections and force law enforcement and judges to treat metadata like data.

Privacy and Dignity

To me, Weitzner validated his role as conference organizer by grounding discussion on basic principles. He asserted that privacy means letting certain people handle data without allowing other people to do so.

I interpret that statement as a protest against notorious court rulings on "expectations of privacy." According to US legal doctrine, we cannot put any limits on government access to our email messages or to data about whom we phoned, because we shared that data with the companies handling our email and phone calls. This is like people who hear that a woman was assaulted and say, "The way she dresses, she was asking for it."

I recognize that open data can feed wonderful, innovative discoveries and applications. We don't want a regime where someone needs permission for every data use, but we do need ways for the public to express their concerns about their data.

It would be great to have a kind of Kickstarter or Indiegogo for data, where companies asked not for funds but for our data. However, companies could not sign up as many people this way as they can get now by surfing Twitter or buying data sets. It looks like data use cannot avoid becoming an issue for policy, whoever sets and administers it. Perhaps subsequent workshops will push the boundaries of discussion farther and help us form a doctrine for our decade.

What's Up with Big Data Ethics?

Insights from a business executive and law professor

by Jonathan H. King and Neil M. Richards

If you develop software or manage databases, you're probably at the point now where the phrase "Big Data" makes you roll your eyes. Yes, it's hyped quite a lot these days. But, overexposed or not, the Big Data revolution raises a bunch of ethical issues related to privacy, confidentiality, transparency and identity. Who owns all that data that you're analyzing? Are there limits to what kinds of inferences you can make, or what decisions can be made about people based on those inferences? Perhaps you've wondered about this yourself.

We're obsessed by these questions. We're a business executive and a law professor who've written about this question a lot, but our audience is usually lawyers. But because engineers are the ones who confront these questions on a daily basis, we think it's essential to talk about these issues in the context of software development.

While there's nothing particularly new about the analytics conducted in big data, the scale and ease with which it can all be done today changes the ethical framework of data analysis. Developers today can tap into remarkably varied and far-flung data sources. Just a few years ago, this kind of access would have been hard to imagine. The problem is that our ability to reveal patterns and new knowledge from previously unexamined troves of data is moving faster than our current legal and ethical guidelines can manage. We can now do things that were impossible a few years ago, and we've driven off the existing ethical and legal maps. If we fail to preserve the values we care about in our new digital society, then our big data capabilities risk abandoning these values for the sake of innovation and expediency.

Consider the recent $16 billion acquisition of WhatsApp by Facebook. WhatsApp's meteoric growth to over 450 million mobile monthly users over the past four years was in part based on a "No Ads" philosophy. It was reported that SnapChat declined an earlier $3 Billion acquisition offer from Facebook. Snapchat's primary value proposition is an ephemeral mobile message that disappears after a few seconds to protect message privacy. Why is Facebook willing to pay Billions for a mobile messaging company? *Demographics and Data*. Instead of spending time on Facebook, international and younger users are increasingly spending time on mobile messaging services that don't carry ads and offer heightened privacy by design. In missing this mobile usage, Facebook is lacking the mobile data. With WhatsApp, Facebook immediately gains access to the mobile data of hundreds of millions of users and growing. While WhatsApp founder Jan Koum promises "no ads, no games and no gimmicks" and has a board seat to back it up, Facebook has a pretty strong incentive to monetize the WhatsApp mobile data it will now control.

Big Data is about much more than just correlating database tables and creating pattern recognition algorithms. It's about money and power. Big Data, broadly defined, is producing increased powers of institutional awareness and power that require the development of what we call Big Data Ethics. The Facebook acquisition of WhatsApp and the whole NSA affair shows just how high the stakes can be. Even when we're not dealing in national security, the values we build or fail to build into our new digital structures will define us.

From our perspective, we believe that any organizational conversation about big data ethics should relate to four basic principles that can lead to the establishment of big data norms:

- *Privacy isn't dead;* it's just another word for information rules. Private doesn't always mean secret. Ensuring privacy of data is a matter of defining and enforcing information rules—not just rules about data collection, but about data use and retention. People should have the ability to manage the flow of their private information across massive, third-party analytical systems.
- *Shared private information can still remain confidential.*It's not realistic to think of information as either secret or shared, completely public or completely private. For many reasons, some of them quite good, data (and metadata) is shared or generated by design with services we trust (e.g. address books, pictures, GPS, cell tower, and WiFi location tracking of our cell phones). But just because we share and generate information, it doesn't follow that anything goes, whether we're talking medical data, financial data, address book data, location data, reading data, or anything else.
- *Big data requires transparency.* Big data is powerful when secondary uses of data sets produce new predictions and inferences. Of course, this leads to data being a business, with people such as data brokers, collecting massive amounts of data about us, often without our knowledge or consent, and shared in ways that we don't want or expect. For big data to work in ethical terms, the data owners (the people whose data we are handling) need to have a transparent view of how our data is being used—or sold.
- *Big data can compromise identity.*Privacy protections aren't enough any more. Big data analytics can compromise identity by allowing institutional surveillance to moderate and even determine who we are before we make up our own minds. We need to begin to think about the kind of big data predictions and inferences that we will allow, and the ones that we should not.

There's a great deal of work to do in translating these principles into laws and rules that will result in ethical handling of Big Data. And there's certainly more principles we need to develop as we build more powerful tech tools. But anyone involved in handling big data should have a voice in the ethical discussion about the way Big Data is used. Developers and database administrators are on the front lines of the whole issue. The law is a powerful element of Big Data Ethics, but it is far from able to handle the many use cases and nuanced scenarios that arise. Organizational principles, institutional statements of ethics, self-policing, and other forms of ethical guidance are also needed. Technology itself can help provide an important element of the ethical mix as well. This could take the form of intelligent data use trackers that can tell us how our data is being used and let us make the decision about whether or not we want our data used in analysis that takes place beyond our spheres of awareness and control. We also need clear default rules for what kinds of processing of personal data is allowed, and what kinds of decisions based upon this data are acceptable when they affect people's lives. But the important point is this—we need a big data ethics, and software developers need to be at the center of these critical ethical discussions. Big data ethics, as we argue in our paper, are for everyone.

Wind Turbine
Profile for 2500 KW Wind Turbine
Miles per Second
SAN JOSE
25 FINSBURY CIRCUS
London, UK EC2M 7EE
LONDON
floor trader 788
LAST TRADE AMOUNT € 9322,50
TOTAL TRADES € 098022078,49
NEW YORK