



Disrupting the Disruptors: Three Design Patterns for Combatting Disruption in Incumbent Organizations

by Dan Gordon

In a world where change and threat come from all sides, from familiar competitors and directions as well as from left field, it behooves all of us to consider how to fend off competitive or substitute threats arising from disruptive technology innovations. While beating disruptive technology threats is very difficult for "incumbents," it is not impossible. In this article, I present three design patterns that incumbents have used to advantage in the past, together with some tips for succeeding with each of them.

CHRISTENSEN 101

As with other "oughta-reads" such as the US Constitution or the Bible, many fine people go through life under the impression that they know what Clayton Christensen's book *The Innovator's Dilemma*¹ is about. Many of us feel that we understand what he meant by "disruption" and "disruptive technology."

Unfortunately, many of us are mistaken. Even a casual review of the term "disruptive technology" shows that most people use it to mean "any technology innovation that causes a change." This is not what Christensen had in mind.

The essence of Christensen's work is that disruptive technology innovation is a particular kind of competitive threat, marked by:

- **Inferior but novel technology** whose initial impact on a market is slight and peripheral
- **Progressive expansion** of the inferior technology to more and more use cases

What marks a disruptive technology, then, is the gradual expansion of a technology initially considered "inferior" for all but a few cases to the point where it eventually covers the market.

Inferior But Novel

Whatever the existing marketplace values — performance, scale, usability, durability, size — is typically

handled well by incumbent technologies. Incumbent vendors are vigorously expanding their products' performance, scale, or what-have-you in response to customer demands.

Calling this kind of market disruption a "disruptive technology" is perhaps a bit of a misnomer. The disruption comes from the market innovation, not from the technology. In fact, the disruptor's technology usually does not maximize performance, scale, and the like but is:

1. "Good enough" with respect to those features
2. (Frequently) cheaper
3. Able to offer a novel benefit that appeals to some customers or some use cases.

In terms of eventual market adoption, it is the third factor that is key. In terms of stymieing incumbents, the first and second factors are more salient.

Progressive Expansion

The successful disruptive technology innovation slowly expands to a larger and larger customer set until it has driven the incumbents out of the original marketplace or driven them into a small niche.

Non-"Christensenian" Disruption

Other kinds of disruption are not of this "Christensenian" sort. There are cases, for example, where a superior technology displaces an inferior one (jet engines versus prop engines for aircraft come to mind, or Google versus AltaVista). Because the technology in these cases is actually superior to that currently in use, these are not disruptive technologies in the sense that Christensen intends, even when the superior technology has growing pains or other shortcomings that initially keep it from broad use.

There are also cases where an insurgent win over an incumbent doesn't have technology at its core. The rise of Southwest Airlines over its incumbent rivals is mainly a disruption based on business model.

WHY CAN'T INCUMBENTS FEND OFF CHRISTENSENIAN DISRUPTIONS?

Part of the horrible fascination of technology disruption of the Christenian sort is the inability of the incumbents in the marketplace to react competitively. Imitating the *technology* aspects of a disruptive technology is generally not a difficult exercise for an incumbent, and some within the incumbent organization argue forcefully for a response. Yet Christensen's work — and our own experience — shows a landscape littered with incumbents whose lunch was eaten by disruptive technology innovations in their marketplace. The reasons have to do with the very inferiority of the innovation and the very gradualness with which it expanded into the marketplace.

"Inferior," of course, is a subjective and relative term and has everything to do with how the customers in the marketplace look at incumbent and disruptive players. It also, as Christensen points out, changes over time. What makes technology disruption work is that customers will consider inferior "specs" if the price is right.

Disruptive technology entrants gain traction in a niche because (1) the cost sensitivity of the niche customers is high, (2) their desire for "state-of-the-art" features is low, and/or (3) the disruptive technology offers a new kind of benefit that is of no interest to the existing market as a whole.

Consider, for example, the rise of the PC, which was disruptive to the incumbent computer solutions (mainframe, midrange, and workstation computer solutions):

- The market valued MIPS (which is shorthand, of course; what the market really valued was computing power and capacity). PCs were deeply inferior in this department and could justly be considered "toys."
- Some customers couldn't afford bigger iron and/or welcomed PCs for peripheral jobs such as word processing, which was being done on dedicated appliances.
- PCs offered a unique application (the spreadsheet) that caused many buyers to want them.

The PC example has been famously worked and reworked by those who study disruptive technologies. DEC made several attempts to build PCs and was utterly unsuccessful. IBM, of course, made the pivot for reasons we will discuss below.

Incumbents are typically unable to fend off disruption from below because:

- Incumbent cultures discourage disruptive innovation.
- Initially, disruptive opportunities are trivially small compared to existing ones; resources won't be allocated.
- Incumbents cannibalize the best features from disruptive teams and add them to the incumbent products.
- Incumbent market feedback systems are weak on detecting disruptive opportunities.

These factors work together. The talent in an incumbent company is geared toward finding large profitable opportunities to expand the market of the incumbent — opportunities, as they say, that "move the needle." The most talented managers, technologists, and product innovators wouldn't be caught dead in a disruptive project, which is "career limiting" or, worse, "career suicide." Customers (at least the customers the incumbent talks to) want more/faster/better, not worse/slower/cheaper. And if a disruptive technology does have a useful new feature, the incumbent incorporates it into an existing product line and sells the enriched incumbent product against it.

Just Saying No to NoSQL

An interesting modern example of this phenomenon is a CACM blog post by Michael Stonebraker — a pioneer of the RDBMS movement and consistent innovator over the years — in which he trash-talks NoSQL:²

According to a recent ReadWriteWeb blog post by Audrey Watters, 44% of enterprise users questioned had never heard of NoSQL and an additional 17% had no interest. So why are 61% of enterprise users either ignorant about or uninterested in NoSQL?

To get more color on the situation, I contacted a very senior technical guru at a large enterprise [and] I asked him how interested he was in NoSQL and, in effect, how interested his company was. He reported "no interest." I asked him why.

He made one comment about OLTP, one comment about warehouses, and one general comment:

- No ACID Equals No Interest
- A Low-Level Query Language is Death
- NoSQL Means No Standards³

In Christensenian terms, Stonebraker is saying that the disruptive NoSQL technologies are deficient in the features the largest enterprise customers of RDBMS technology care about, and are therefore useless to them.

The core NoSQL technology at which Stonebraker is directing his fire is probably MapReduce/Hadoop, the

“big data” approach Google developed for distributed processing of unstructured data. MapReduce processes data in files in a distributed “map” phase of simple but very parallel operations on the data, followed by a “reduce” phase to perform more complicated operations on an aggregated subset of the data. Unlike an RDBMS approach, this approach does not maintain ACID (atomicity, consistency, isolation, durability) guarantees on data operations and uses standard application languages for the map and reduce operations rather than a query language. As a good disruption should, MapReduce has advantages for certain workloads (easy incremental scalability and rapid application development), which cause take-up in circumstances where the adopters don’t mind the disadvantages (no ACID, no SQL). As one commentator says on Stonebraker’s blog:

Funny — I would take the same numbers and draw the opposite conclusion.

39% of enterprises ARE interested in nosql [sic]. Considering the nosql [sic] products themselves have only said they’ve been ready for a year or so, getting 39% of enterprises interested in that time is kind of amazing.

— Dan Lync, 1 October 2010⁴

No student of Christensen can read this discussion without concluding that:

- Stonebraker is voicing the full-throated bull challenge of an incumbent technology.
- NoSQL isn’t going to take over the OLTP core of the incumbent franchise any time soon.
- If pattern matching on trash talk is any indication, the RDBMS may yet become a victim of disruptive technology innovation from the NoSQL movement.

While it is not *impossible* for market incumbents to avoid being disrupted by technology innovation “from beneath,” it is very difficult. Competition from a disruptive technology is not necessarily a death sentence, but it is a serious illness that needs radical treatment. The problem is that incumbents are seldom willing or able to take the treatment.

DESIGN PATTERNS THAT MAY WORK FOR INCUMBENTS

Design patterns are abstract descriptions of a solution to a class of problems. The term was coined by architect Christopher Alexander⁵ in an attempt to express timeless patterns that recur in successful architectural design projects, and the concept has been enthusiastically adopted by the IT architecture community.⁶ Here are three design patterns for measures that can successfully combat disruption.

Pattern 1: Stealth Skunkworks

Pattern Synopsis

In the Stealth Skunkworks pattern, an incumbent “forks off” a group to work on the disruption killer. Whether the group is actually spun off or merely virtually distinct, it is vital that the group not interact with the main body of the incumbent organization.

Case Study: Original Macintosh, 1984

Steve Jobs’ hijacking of and direction for the Apple Macintosh team is a textbook example of Stealth Skunkworks. Prior to Jobs’ involvement, the Macintosh team was working on a next-generation product for Apple along the lines of a “cheap people’s computer.” Led by Jef Raskin, the team was on its way to producing something a bit like the “hundred-dollar laptop” of 10 years ago.

Jobs, who had been somewhat marginalized as a leader of the company, had a very different vision: to create a computer that used the GUI from Xerox to make a “creative appliance” for knowledge workers. His instinct was to seize control of the Mac group — Raskin left Apple after losing many struggles with Jobs — and move it to a separate building, with a separate team mystique and identity, isolated from the mainstream of Apple.

Within Apple, Jobs gathered a small, conniving team — and he did not care for other projects in the company.... Above the roof of “Bandlely III” [the building, remote from Apple’s executive suite, where Jobs housed the Mac team], a pirate flag with the Apple symbol as eye patch was waving — and on deck of the virtual pirate ship, Steve Jobs was standing as a man who wanted to prove it to them all.⁷

Case Study: Birth of the IBM PC

To develop the PC, IBM set up a separate subsidiary with separate leadership, reporting structure, metrics, and goals in a non-core IBM location: Boca Raton, Florida. Don Estridge, who led the IBM PC project, made a collection of decisions that were decidedly not in keeping with mainstream IBM culture:

There were a number of unique aspects of the IBM PC that made it different from previous IBM products. In designing the IBM PC, the team at IBM elected to use off-the-shelf parts rather than the usual IBM-sourced components. In another unconventional decision, they publicly released the basic hardware specification to encourage other companies to manufacture and sell IBM-compatible boards. Last of all they decided not to exclusively provide the software applications for the product, but instead to encourage third parties to develop software for the PC

and license it directly to customers. Except for a critical piece of code called BIOS, the PC truly was an open hardware architecture. Every major and minor component, enclosures, motherboards, disk, memory, bus, even the CPU would eventually be easily second-sourced.⁸

Success Notes

Many things about the Stealth Skunkworks pattern are unclear, but the central mystery is this: Does the team *have* to be separated from the rest of the organization, and, if so, why? I believe these are some of the factors that make separation of the Stealth Skunkworks team a necessity:

- **To keep incumbent product teams from cannibalizing cool features.** Incumbent teams have lots of clout in a successful organization, and they can grab features — or, worse, developers — away from the disruptive product.
- **To insulate disruptive team members from career consequences.** What Jobs did for the Macintosh team was to personally underwrite insurance on their career risk and protect them from worrying about it by preempting all lines of communication between them and the rest of Apple.
- **To permit the disruptive team to brainstorm freely about a disruptive product.** What makes a successful disruption is not just a disruptive technology but a complete product built on that technology. Siebel's mistake was to build a Salesforce competitor that essentially skinned the Siebel product — with all of its feature-heavy bulk — with a SaaS wrapper rather than reimagine what a SaaS-based product might be and identify customers for whom it might suffice.

Pattern 2: Defcon 1 Pivot

Pattern Synopsis

In the Defcon 1 Pivot pattern, the leader of an incumbent organization declares by fiat that the organization will transform itself and executes a draconian shift in direction from the top down.

Case Study: Microsoft Internet Tidal Wave, 1995

A classic example of this is Bill Gates' "Internet Tidal Wave" memo of the mid-1990s and the change in Microsoft's direction he brought about in a relatively short period of time.

In May 1995, when Gates wrote his memo, it was the year that the Netscape IPO would galvanize Silicon Valley into the realization that the Internet — and Internet-based businesses and technologies — were

going to be the Next Great Thing. Microsoft at the time dominated the PC desktop market and was rapidly gaining ground in the server market (in itself a disruptive technology change). Marc Andreessen, one of Netscape's founders, famously disparaged Windows as a "poorly debugged device driver," meaning, among other things, that the PC as a serious node for computing innovation was a dead man walking.

Web-based software was at the time a disruptive technology: easier and cheaper to develop than client-server technology, but inferior. The long latencies and spotty network performance, as well as the impoverished UI resources of HTML, guaranteed that software based on server-side applications would be perceived as slow, ugly, and unreactive by users accustomed to client-server applications. Microsoft had every reason to sneer at the innovation and rest in the fortress of the desktop until the company was completely irrelevant.

Gates did no such thing. His long memo went into detail about how the Internet was seen by Microsoft customers, how it might develop, and what this would do to Microsoft's then-current business. Here is a retrospective gloss by Dylan Love, a *Business Insider* blogger:

Called "The Tidal Wave of the Internet," the lengthy document went into minute detail about what the Internet is and how Microsoft could exploit it.

Our favorite quotes appear below:

"One scary possibility being discussed by Internet fans is whether they should get together and create something far less expensive than a PC which is powerful enough for Web browsing."

"After 10 hours of browsing, I had not seen a single Word .DOC, AVI file, Windows .EXE (other than content viewers), or other Microsoft file format. I did see a great number of QuickTime files."

"Amazingly it is easier to find information on the Web than it is to find information on the Microsoft Corporate Network."

"For now, Acrobat files are really only useful if you print them out, but Adobe is investing heavily in this technology and we may see this change soon."

"I believe the Internet will become our most important promotional vehicle and paying people to include links to our home pages will be a worthwhile way to spend advertising dollars."⁹

Gates followed up on the memo with relentless top-down pressure to reform Microsoft's core businesses. Microsoft, in short order, refashioned its online property MSN to focus on the Web and vigorously boosted Internet Explorer, which was bundled with the Service Pack 1 release of Windows 95. These were significant moves, but the heart of the Defcon 1 Pivot pattern is

not necessarily the concrete tactics but rather the urgency with which the priorities of the organization are rearranged. Microsoft had built TCP/IP support into the heart of Windows 95 before Gates's memo, which may well have been the move above all others that propelled the World Wide Web into hypergrowth.

A number of Microsoft's practices with IE got the company into anti-trust trouble, but the boosting of IE was the correct strategy for Microsoft despite some sketchy tactics. It should also be noted that IE's success had a lot to do with perfectly sound tactics as well as the sketchy ones. (We might further observe that, today, Internet Explorer is itself being disrupted by insurgents Firefox — whose lineage goes back to Netscape — and Chrome, which focus on speed and security against IE's bevy of features.)

Other Cases

- Lou Gerstner led a famous Defcon 1 Pivot at IBM in the mid-1990s, refocusing the company away from siloed groups and hardware and toward customer focus and software/services. IBM was challenged here by a long-term technology disruption: the growth of "scale-out" models of hardware deployment, where large clusters of commodity servers (and, more recently, commodity storage devices) are initially unable to match mainframe performance but offer easy scalability and enable expansion of the system at inexpensive, incremental cost. Eventually the scale-out systems caught up to the "scale-up" systems and destroyed IBM's business. So as margin shifted to software and services, Gerstner manhandled IBM into the future.
- Sun tried a similar but much less successful Defcon 1 Pivot in the Jon Schwartz years, veering away from its hardware business — which was being decimated by the dot-com collapse — to a focus on software and services. Unfortunately for Sun, the scale-out server disruption was much more fatal to the company's position as the engine of the dot-com revolution than it was to IBM's mainframe and departmental hardware business, and Sun was not able to hold up its revenues or earnings during the course of the pivot.

Success Notes

Judging from the Microsoft example, here are a few essential elements of the design pattern:

- **The call to action for the organization must be global.** Gates said that every Microsoft business unit had to rethink its actions based on the Internet. In point of

fact, an exception was made for Windows 95, due to ship that summer, but only a temporary exception. Every part of the Microsoft business was required to react. Limiting the call to action to a particular team or group would have been the kiss of death.

- **The pivot must contain an element of self-criticism.** Without leadership backing the idea that there were shortcomings in the previous course of action, managers won't have a reason to change. They will claim that they were following — or at least advocating — the new direction all along. Self-criticism at the top allows everyone to say, "OK, we were short-sighted before. Here's what we're going to do now."
- **Leadership needs to follow up with detailed assistance to business units in executing the pivot.** Gates's main activities following the memo were leading and reviewing plans for detailed business unit change.

Pattern 3: Acquire a Disruptor

A feasible pattern, although seldom done successfully, is to acquire a disruptive product line directly. Cisco executed this pattern brilliantly during the 1990s and early 2000s but has been unable to replicate its success as a nondestructive acquirer more recently. Its decline has much to do with internal issues, to be sure, but also with the nature of the companies Cisco is trying to acquire. Its dot-com acquisitions were largely disruptive network technologies that Cisco understood well; the new acquisitions have been in noncore businesses into which Cisco wanted to expand but did not understand as well.

Pattern Synopsis

The incumbent organization does a formal acquisition of a disruptive competitor or substitute, and then manages it somewhat like a Stealth Skunkworks by:

- Isolating the acquired company's team
- Allowing it to retain its own culture, approach to market, and even marketing and sales resources
- Slowly figuring out the implications of the acquired company's approach for the wider organization

Acquisitions are often done in the hopes of acquiring a disruptor, but the devil is in the details.

Case Study: Intuit's Acquisition of Mint.com

Intuit's acquisition of disruptive competitor Mint is an ongoing example of an attempted Acquire a Disruptor

play. Mint, launched in 2007, was a disruptive competitor to Intuit's Quicken, which dates from 1984. Quicken, in classic incumbent fashion, is a massive collection of user-demanded features in a somewhat ponderous desktop-client configuration that is (more) difficult than mint.com to deploy, install, and upgrade.

In contrast to Quicken, Mint is:

- A pure Web app and so does not require user download or upgrade. (Frequent "forced" upgrades are a frequent Quicken user complaint.)
- Strongly oriented toward *recording financial transactions quickly* (the original goal of Quicken, by the way; hence the name) rather than elaborate reporting, planning, categorizing, or other derivative uses of the financial information. (The "confusing" variety of features is a common Quicken user complaint.)
- Free, sponsored by financial services organizations that pay for the leads generated through analysis of users' habits and vendor offers.

In 2009, Intuit and Mint (which then had 1.5 million users) agreed to an acquisition of the latter by the former.

Intuit had started a number of internal "Quicken killer" projects over the years, and some of them had a similar profile to Mint's, but perhaps for the reasons discussed above, Intuit was unable to bring them to market. The company's hope in the acquisition was, evidently, that Mint would survive where the internal projects had failed.

Intuit has put Mint CEO Aaron Patzer in charge of a division managing both Mint and the existing Quicken product family, possibly a questionable move in light of the Success Notes below.

Case Study: NewsCorp's Acquisition of Myspace

NewsCorp's disastrous acquisition of Myspace has to count as a strong example of a failed Acquire a Disruptor strategy. In 2005, NewsCorp was astute enough to notice that both its audience and its advertising revenues were being drawn to the Internet by disruptive innovation. They were also astute enough to see that the key acquisition on the Web might not be a news property but a property with a large and growing audience. Myspace fit the spec.

The scheme was to run Myspace as an autonomous unit, continuing with its success formula, but over time the incumbent drew it in more and more closely while distracting the Myspace team with broader NewsCorp

responsibilities. The company was hoping — as is the case in many Acquire a Disruptor plays — to apply the Myspace "magic" to other NewsCorp properties. The result? Myspace became bogged down in endless "build-not-buy" projects (which the Myspace principals might have been able to sort through if they had not been distracted) and ultimately wilted in the face of rising pure-play audience rival Facebook. It was sold for scrap to Specific Media in 2011.

Success Notes

In this pattern, some of the same rules as Stealth Skunkworks apply, in a modified form:

- **Do not allow the incumbent culture to co-opt the disruptor culture.** Keep the disruptor group as a kind of external Stealth Skunkworks, or use its management to shake up the incumbent business units instead of vice versa. (Although this, too, is risky; see below.)
- **Set separate goals for the disruptor products.** A disruptive product line that is all of a sudden held to incumbent standards for product growth, share, and/or margin will likely be unsuccessful. The organization will have to have a dual set of metrics for some time.
- **Don't strangle the disruptor with too many resources.** This is a counterintuitive idea, since much of the rationale for acquiring a disruptor is to apply the greater resources of the acquirer to the successful business model of the acquiree. What happened to Myspace was that the abundance of resources suddenly available to them deafened their organizational "ear" for market signals, and they sunk under a welter of projects and features that were not of interest to their audience. The latter jumped ship to Facebook, which at that time focused well on core issues for its audience (safety, communication, community).
- **Don't count on the disruptor to reform the incumbent.** A common mistake is to use the management of the disruptor acquisition to attempt a remake of the incumbent, or of a division of the incumbent. For the same reason that the disruptor organization can be drowned by too many resources, the disruptor leadership can be drowned by too much responsibility and lose their focus on doing what they do best.

CONCLUSION

Incumbent organizations — whether they set up their own Stealth Skunkworks, Acquire a Disruptor, or

attempt a Defcon 1 Pivot — have a tendency to look at a disruptive technology and, more importantly, its market as badly designed versions of their own. “Sure,” they say, “it’s nifty that these [disruptors] have been able to do so well, even with their limited technology resources and laughable processes. Wait until we graft our resources and our processes onto their organization!”

Wrong. This is like cutting open the goose to get at the golden egg. The niche sub-markets that initially respond to disruptive technologies are, in essence, very undifferentiated new markets. When the Macintosh came out, for example, the established PC market was basically a market for spreadsheet applications. The Mac had to discover that desktop publishing was its “natural” new home, a sub-market within the PC marketplace that hadn’t existed before there was a tool that was adapted for it. (Even then, that market didn’t begin to develop in earnest until there was an output device — the laser printer — and middleware — PostScript — that together make up a complete solution.) New markets are ill-defined and delicate things. It takes an adaptive, experimental organization with no hostages against fortune to navigate them.

Myspace had hit on the device of hosting websites for musical groups. It is quite possible that it would have recognized in time that the interests and tastes of its audience were broader than music sites, and that in fact the members of its audience were mainly interested in talking to one another and sharing photos. This is what Facebook realized, and this is what Myspace — engorged with and distracted by NewsCorp resources and processes — was unable to see in time.

What Microsoft realized by doing a Defcon 1 Pivot around the Internet was that the Internet was going to be a part of all computing going forward. Netscape failed, in part, because it thought the Netscape browser and the Netscape portal were going to be the center of the universe, and the universe wasn’t quite ready for that.

What any disrupting-the-disruptors success requires is this: capture one of the new organisms, but then make sure it doesn’t die in captivity.

ENDNOTES

¹Christensen, Clayton M. *The Innovator’s Dilemma: When New Technologies Cause Great Firms to Fail*. Harvard Business School Press, 1997.

²NoSQL is the *cri de guerre* of the movement to replace RDBMSs with “postmodern” data manipulation architectures such as MapReduce.

³Stonebraker, Michael. “Why Enterprises Are Uninterested in NoSQL.” *BLOG@CACM*, 30 September 2010 (<http://cacm.acm.org/blogs/blog-cacm/99512-why-enterprises-are-uninterested-in-nosql/fulltext>).

⁴Stonebraker. See 3.

⁵Alexander, Christopher, Sara Ishikawa, and Murray Silverstein. *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, 1977.

⁶“Design pattern (computer science).” Wikipedia (http://en.wikipedia.org/wiki/Design_pattern_%28computer_science%29).

⁷Dernbach, Christoph. “Steve Jobs Discovers the Macintosh Project.” *Mac History* (<http://www.mac-history.net/the-history-of-the-apple-macintosh/steve-jobs-discovers-the-macintosh-project>).

⁸Koenig, John. “The History of the IBM Personal Computer.” *Compute Magazine*, 2 March 2011 (<http://computemagazine.com/the-history-of-the-ibm-personal-computer>).

⁹Love, Dylan. “This 1995 Memo From Bill Gates Predicts Smartphones, Web Videos, and Internet Ads.” *Business Insider*, 28 July 2011 (http://articles.businessinsider.com/2011-07-28/tech/29989218_1_web-videos-acrobat-files-home-pages).

Dan Gordon is Director of Research at Valhalla Partners, a venture capital firm in the Washington, DC, area. Mr. Gordon has over 30 years’ experience working with technology, as a computer scientist, software developer, manager, analyst, and entrepreneur. Prior to joining Valhalla Partners, he was a Director and senior staff member at the PricewaterhouseCoopers (PwC) Global Technology Centre, analyzing technology trends and consulting on technology-oriented strategies in the software, e-business, wireless, optical, networking, semiconductor IP, and life sciences arenas. Mr. Gordon has worked with clients from North America, Europe, the Middle East, and Australia. He was a Contributing Writer and Contributing Editor to the Technology Centre’s annual Technology Forecast and a frequent speaker at industry and general business meetings.

Prior to joining PwC, Mr. Gordon spent 20 years in Silicon Valley as a software technologist, manager, director, and entrepreneur, including senior technical roles at well-known Silicon Valley firms such as Symantec, Intuit, and Oracle. He has also been involved in startup companies in the applied artificial intelligence and Web applications fields. Mr. Gordon has a bachelor of arts cum laude from Harvard University and master of science degree from New York University in computer science. He is a Professional Member of the ACM. Mr. Gordon lives in Washington, DC, with his wife and two children. He can be reached at pipik1199@gmail.com.