Introduction: Computing's Infrastructural Moment

Jean-François Blanchette

Jay: It went up! It went up to the Cloud!
Annie: And you can't get it down from the Cloud?!?
Jay: Nobody understands the Cloud! It's a fucking mystery!
—From the movie *Sex Tape,* 2014

The interesting thing about cloud computing is that we've redefined cloud computing to include everything that we already do. I can't think of anything that isn't cloud computing with all of these announcements. … Maybe I'm an idiot, but I have no idea what anyone is talking about. What is it? It's complete gibberish. It's insane. When is this idiocy going to stop?
—Larry Ellison, 2008

Here, there, and everywhere.
— Apple iCloud slogan, 2014

From the perspective of policy analysis, cloud computing presents itself as an object with particularly problematic boundaries. Historically, it appears either radically new or merely a reenactment of a bygone computing era, that of mainframes; spatially, it is present in every mobile device, yet highly concentrated in out-of-sight bit processing plants; and at the design level, it seemingly encompasses, as Larry Ellison has noted, every dimension of modern computing architectures. Indeed, depending on one's mood and perspective, cloud computing can be variously characterized as

- The evolutionary end point of computing and the realization of its promise as *utility,* the freedom to access processing power and storage as instantaneously and flexibly as electricity, water, and gas (Armbrust et al. 2010).
- A new era in the *economics* of computing: together with the virtualization of processing and storage, the economies of scale afforded by data centers

allows providers to price computation at an unprecedented granular level, transforming fixed capital investments into operational expenses and lowering barriers to computing innovation (Weinman 2012).

- After the age of personal computing and its concomitant need to manage the proliferation and synchronization of individual devices, applications, and files, a return to the mainframe era and its model of *centralized control* of resources (Lanier 2010).
- The development of a new industrial form, the *data center*, devoted to the efficient transformation of electrical power into flows of bits: data centers further remove computation from the local experience of users, shifting its material constraints to massive bit factories strategically located in areas with access to cheap power and cool weather (Barroso and Hölzle 2009).
- The computing architecture appropriate to the era of *Big Data/Big Brother*: large-scale data centers make possible real-time statistical processing of the troves of data collected by governments, businesses, and scientists—data that will dramatically increase their capabilities for surveillance, targeted advertising, and research (Mosco 2014; Clarke et al. 2013).
- The computing architecture appropriate to the *Internet of Things*: a new paradigm for designing and assembling distributed systems that fully integrates processing, networking and storage resources, free of the historical divisions of computing and telecommunications (Roscoe 2006)
- The elevation of *access to broadband* as a key economic issue for governments all over the world, as well as the defining material line between the digital haves and have-nots (FCC 2010).
- Another illustration of the power of *modular design*, whereas the enormous processing and storage capacity of data centers is based on the interfacing of thousands and thousands of small-bore individual servers and disk drives, themselves often housed in modular architectural structures—for example, shipping containers (Barroso and Hölzle 2009).

As the chapters in this volume will attest to, cloud computing is all of these things at once and more. Indeed, rather than engaging in the thankless task of unifying the above into a comprehensive functional definition, I propose instead that “the Cloud” is shorthand for the moment where computing has become, both materiality and symbolically, *infrastructure;* that is, a sociotechnical system that has become ubiquitous, essential, and foundational (Edwards 2002, 187).[1] As infrastructure, then, the Cloud necessarily becomes the focus of a series of policy concerns that deal with issues of market regulation, fairness, universal access, reliability, criticality, national security, sharing of limited resources, congestion, inter-network competition, national economic welfare, capacity planning, monopoly, and antitrust, among others.

These issues and debates are familiar: they have, in one form or another, featured in every type of energy, transportation, and communication network deployed in the past.[2] Indeed, like the networks that preceded it, the Cloud develops, operates, and breaks down following specific infrastructural dynamics.[3] It has, for example, developed incrementally, from the progressive laying down of its infrastructural components, including data centers, fiber cables, economic models, and regulatory frameworks. Such incremental development means that early-stage design choices *persist*, often with unforeseen consequences, and become increasingly difficult to correct as the infrastructure becomes ubiquitous, its functionality expands, and the nature of the traffic it serves evolves—for instance, the Suezmax standard for shipping, the maximum number of addresses allowed by the IPv4 Internet Protocol and the best-effort service it provides in the context of the dramatic expansion of streaming video traffic. In addition, like other infrastructures, the Cloud strives toward invisibility. Indeed, as the unlucky protagonists of *Sex Tape* bemoan, by staging the very disappearance of computing resources and whisking them away to far-off data centers, the Cloud introduces yet another level of opacity to information technologies.

Yet the Cloud is also distinctive on several fronts. If energy, transportation, and information infrastructures tend to be tightly intertwined, the Cloud's capacity for real-time measurement and statistical analysis of supply and demand makes it increasingly integral to the functioning of a large number of other infrastructures, including energy (smart grids), financial services, airports, the upcoming driverless cars, and even … the Cloud itself.[4] It has become, in effect, a certain kind of *meta-infrastructure*, while of course remaining entirely dependent on the electrical grid. And while all infrastructures are incremental in nature, developing by the gradual layering and interconnection of similar components, the computing infrastructure has elevated this to a quasi-religious principle, with scholars such as Lessig (2001), Zittrain (2008), and Wu (2010) arguing that the innovative character of the Internet is a direct outcome of its modular design structure and the kind of markets this structure supports.

Perhaps most distinctively, the Cloud is expected to sustain extraordinary growth rates. Internet traffic increased eightfold between 2006 and 2011 and is expected to continue growing at an annual rate of 30 percent, with mobile traffic now accounting for 45 percent of all IP traffic.[5] The amounts of data globally generated by scientific instruments, sensors, systems, humans, digitization of legacy documents, and other sources have likewise exploded.[6] There is no reason to expect such rates of growth will taper off anytime soon: ultra-high-definition television (UHDTV) sets will soon match the resolution level of IMAX, while old and new classes of Cloud-based applications (e.g., massively multiplayer online role-playing games [MMORPGs], file backup services, and video surveillance) will make new and unprecedented demands on processing, storage, and network resources. Indeed, there are few practical limits to the quantity of digital information that can be created—only economic and material limits to how much can be stored, processed, and circulated.

Ironically, these extraordinary rates of growth mean that, just as it strives for invisibility, the computing infrastructure is constantly in our face. From legal battles around net neutrality, to slow-streaming videos, to the constant shuffle of new pricing for data plans, the computing infrastructure refuses to settle down as infrastructure, reminding us that, even as we are becoming inexorably dependent on it, we have yet to work out the conditions under which to accommodate such growth.

The chapters in this volume lay the groundwork for analyzing the Cloud as a familiar infrastructure whose distinctive characteristics require fresh approaches to long-standing policy debates. To distinguish the new from the old, such an analysis must begin by situating the Cloud in historical terms. This is complicated, however, by a persistent mythology that presents computing as a field that, as Matthew Fuller (2008, 7) puts it, needs no history, as it relentlessly moves forward, "permanently in a state of improvement." Yet the Cloud is the outcome of distinct evolutionary processes that hark back to the early beginning of computing, a history that is essential to understanding current policy debates. I begin by providing some of that historical background, so as to set the stage for the following chapters, each of which will address specific policy issues.

The Historical Genesis of the Cloud

The evolution of computing technology is typically portrayed in terms of its extraordinary rates of growth in performance: year after year, processors perform more instructions at ever-increasing speeds, storage devices are able to pack more bits into ever-smaller amounts of space, and network wires transmit more data at ever-faster rates. Indeed, much of our understanding of the extraordinary spread of computing in the past 60 years is based on the idea that the fundamental computing resources of processing power, storage, and bandwidth have and will continue to become simultaneously more powerful and cheaper.[7]

This particular frame of analysis, however, captures only a limited subset of the forces that have driven the evolution of networked computing. Less visible but equally essential dimensions include, on the one hand, the design technique of *modularity*, used to manage the high rate of technological change that characterizes computing technologies; and on the other hand, the *sharing* and *distribution* of limited computing resources. Much of these dynamics take place somewhat out of sight, at the level of the *computing infrastructure* rather than at the level of applications, the more plainly visible space where users extract personal value from computing technologies. Yet, it is only by taking together these three shaping forces—performance increases, modular design, and sharing—that the evolutionary dynamics of the computing ecosystem, including its current manifestation as the Cloud, can be analyzed.[8]

The Computing Infrastructure

In computing, infrastructure can be quite simply defined as the elements of the computing ecosystem that provide *services to applications* (e.g., performing arithmetic functions, storing and retrieving bits, sending packets over networks), in contrast to the applications that provide *services to users* (e.g., processing words, posting a status update). The computing infrastructure is composed of both software and hardware: for example, system abstractions such as file systems and packets, storage media such as flash and hard drives, and communication protocols such as Transmission Control Protocol/Internet Protocol (TCP/IP) and the Hypertext Transfer Protocol (HTTP). It provides the various components from which designers can build computing systems as diverse as the Google search engine, Microsoft Word running on a desktop, and Tivo software running on your TV set. It is what allows computers to be *multipurpose* while simultaneously *managing the high rate of technical change* of computing resources. These two characteristics of computing systems seem largely obvious today, but they required considerable design innovation in their own time.

The first electromechanical computing machines could perform only a limited range of computational tasks. Herman Hollerith's first tabulator, for example, developed for the 1889 US census, was specifically tailored to add up census schedules. Early digital computers, such as the ENIAC, could be reconfigured to perform different types of computation only through a time-consuming rewiring of the various hardware components. The concept of the *stored program*, as formulated by John von Neumann and colleagues, provided one elegant solution to the issue: the activation of the different components of computers in the service of a computational task could be directed by a sequence of instructions called a *program*. A single computer could perform distinct computational tasks merely by executing a different program, and while designing such programs proved to be no simple task, once written, a program could be switched for another one in a matter of seconds.

Modularity

This newfound versatility came with some important trade-offs, however. Even as von Neumann and his co-inventors gave birth to the software/hardware division, the two remained fused: in the early days of computing, there is hardly any distance between programs and hardware. The manual of operations for the first commercially produced stored-program computer, the IBM 701, included such specific timing considerations as "to keep the card reader in continuous motion, it is necessary to give the READ instruction between 20 and 70 milliseconds after the 12-right COPY instruction" (International Business Machines Corporation 1953, 49). That is, programmers had to take into account specific characteristics of the hardware—in this case, the number of operations that the processor would execute before the next card would be available for reading data or instructions from the punched card reader (the dominant input/output technology in the early 1950s). This became rapidly problematic, insofar as each succeeding generation of computers offered new, faster hardware, and programs had to be rewritten from scratch to take advantage of their new characteristics, at great expense of time and money.

By the 1960s, the situation had become a sore spot for the industry as a whole, prompting market leader IBM to seek a remedy. The solution consisted in designing a family of processor lines (initially six), each compatible with one another (including peripherals); that is, any program written for one line would be able to execute on any other of the family, with, of course, different levels of performance. To achieve such compatibility, IBM relied on the design strategy of *modularity*, where each component of the system was conceived as a discrete black box with a standardized *interface* (Baldwin and Clark 2000). In the resulting System/360, for example, all processors responded to the same set of instructions, even though their internal architectures might differ widely. For the first time, programmers could design programs with the confidence that, as long as components retained the same interface, programs would continue to run in spite of technical advances.

Compatibility proved much more than just an engineering feature, however, as it profoundly altered the economics of the computing ecosystem: components with standardized interfaces could just as well be produced by competitors, and many IBM engineers did leave the company to launch their own lines of cheaper compatible processors and peripherals. Modular computing systems rapidly ushered in an era of vertical disintegration of the industry and a new form of market organization. Indeed, innovation scholars such as Lessig (2001), Zittrain (2008), and Wu (2010) have since argued that the extraordinary success of the Internet is a direct outcome of its modular architecture, which effectively lowers barriers to entry for prospective market participants and fosters experimentation at the component level.

Sharing and Virtualization

Another parallel historical development involved the development of *operating systems* and *virtualization*. In the early days of computing, computing speed was hampered by two main bottlenecks: on the one hand, programs were loaded and executed sequentially in a slow and cumbersome process called *batch processing*: at any one time, only one user could access the machine's expensive resources, such as its processor or storage media. On the other hand, increases in processing speed were limited by the extremely slow speed of storage technologies: program execution was often stalled as the processor waited for data to be loaded or written to storage.

A decisive breakthrough came with the invention of *time-sharing*. Instead of executing sequentially, multiple programs were loaded simultaneously and executed under the authority of a new program, the *supervisor*. In a similar fashion to time-sharing in real estate, the supervisor amortizes an important capital expense (the processor) by distributing it among multiple noncompeting users. By allocating to each program a slice of processing time and by circling rapidly in round-robin fashion among them, each user was given the illusion of having full control of the computer's resources, while in actuality having that control for only a small fraction of the time. Because the supervisor could use the time previously spent waiting for storage devices to service other users, individual performance did not suffer overall.

In today's terms, time-sharing *virtualized* the processor: it created an *abstraction* of a computing resource—a whole processor, when only a portion was available—so that it could be more efficiently shared among multiple users. User programs no longer directly interacted with the processor, but with the abstraction provided by the supervisor, which sliced and diced the actual processor among as many users as could be supported.

Just like modularity, time-sharing profoundly changed the economics of computing: by allowing more users to share the most expensive component of a computer (its processor), institutions were able to extract more usage out of their more expensive capital investments. Through the design of appropriate abstractions, enormous gains in computing efficiency could be obtained by the efficient *sharing* of costly and limited computing resources. At the same time, the supervisor ushered in the era of *operating systems*: software that would serve as a *mediating layer* to manage applications' access to computing resources, whether processor, storage, or network.

Distributing Computing

Another strand of the history of computing design can be understood as addressing the problem of the *distribution of computing resources in space*. That is, where should computing resources (i.e., processing, storage, and data) be located in relationship to each other and to users? The criteria for choice includes not only computational efficiency, but also crucial issues of control, cost, maintenance, reliability, security, and access, among others. During the relatively short history of computing, different architectures have successively dominated the landscape.

Early digital computers took the shape of *mainframes*: single-user machines where all computing resources were centrally located, controlled, and maintained, often in the same room. Users accessed the mainframe through computer operators who controlled available software and data, both of which had to be loaded on input/output devices located in the same physical space.

The advent of *time-sharing* dramatically transformed this setup: the virtualized processor of the mainframe was partitioned among multiple users, who could access it through terminals (connected either through local wires or phone lines). The mainframe functioned as a *server* to these multiple *clients*, providing access to software and data, stored either locally or remotely. At this stage, the data traveling over the wires was textual: commands typed by users and the results of their queries, typically textual and numeric information contained in databanks or the output of programs. While access was expanded, control remained local to the machine. Security was a new problem, as multiple users shared access to processors, storage, data, and programs.

Personal computing yet again expanded the reach of computing by providing users with unprecedented control over their own processors, storage devices, software, and data. At the same time, it introduced a host of challenges in the workplace: in contrast to a centralized mainframe, every employee's computer needed to be set up, maintained, upgraded, repaired, and provided with individual copies of software. Also, processing and storage capacity were potentially wasted, as individual machines sat idle at night. Personal computers also integrated with mainframes, insofar as they could be used as terminals to connect to institutional mainframes to access software or commercial services (e.g., databanks).

By allowing personal computers to connect to one another, the Internet yet again broadened the scope of architectural possibilities for the distribution of computing resources. By and large, the dominating relationship has been one of client/server: users' devices download content (e.g., query results or streaming data) from the Cloud to local clients (primarily web browsers), with Netflix as the current paradigmatic example. This is, however, only one of many possible configurations for distributed computing: peer-to-peer computing, in its many flavors, provides a vibrant example of an entirely different model for pooling together distributed resources, one whose applications go well beyond mere illegal file sharing; and grid computing, as exemplified by the SETI@Home project, leverages the idle cycles of thousands of machines to solve computationally intensive problems, such protein folding or climate modeling.

Today's age of *mobile computing* has emerged in symbiotic relationship with the Cloud. Given their limited storage, processing, and energy resources, portable devices such as smartphones, tablets, and netbooks rely on cloud services to provide the required software capabilities for the mobile services users have come to rely on, such as maps and voice recognition. This movement of processing cycles and storage away from users' machines towards data centers depends entirely, however, on the availability of *broadband*. The more data-intensive the service, the more bandwidth required. For many classes of applications (e.g., video editing), bandwidth requirements and availability continue to make cloud-based processing an unattractive option.[9] Indeed, data-intensive services such as Second Life and Google Earth must be accessed through client software (or web plug-ins) that can render three-dimensional virtual environments with the help of a device's local processor. Furthermore, in the cloud model, reliability is entirely a function of service providers, which themselves depend on network service providers—we might say, "live by the Cloud, die by the network."

Moreover, the centralization implied by data centers is already being mitigated by content distribution networks (CDNs) such as Akamai, designed to move resources (processing, storage, and data) closer to the edge of the networks, that is, users' machines. Large content providers like Netflix are developing their own private CDNs (e.g., Open Connect) to bypass the costs and network congestion that results from moving large amounts of data across the Internet. And of course, data centers are themselves being strategically located to minimize not only data movement, but also energy costs. Even in a cloud-based world, then, multiple models remain for the distribution of computing resources, whether by reason of computing efficiency or political conviction, and spatial distribution of these resources continues to matter.

Cloud Shapes

The Cloud thus emerges at the historical confluence of several long-standing technical traditions within computing: *modularity*, which has allowed cloud providers to create unprecedented amounts of computing power by merely pooling together massive numbers of low-cost, off-the-shelf components; *virtualization*, which makes it possible to distribute, meter, and charge for these computing resources in highly granular and flexible ways while allowing continuity with legacy software designs; and *distributed architectures,* which allow for the partitioning of computing resources between mobile devices and data centers.

Within these broad characteristics, several categories of cloud models have emerged, not only in terms of the types of services offered to customers, but also deployment models (e.g., on site or outsourced), as well as different levels of risk, operational characteristics (e.g., performance and reliability), terms of service, and economic considerations. Of primary importance is the *control* that consumers and providers exert over different types of cloud computing resources—application software, operating system, hardware, etc.[10]

The first category deals with that manifestation of the Cloud familiar to most Internet users through the services provided by, for example, Google (i.e., Gmail, Google Docs, Google Maps, and Google Drive). In such an arrangement, usually referred to as *Software as a Service (SaaS),* users interact with the Cloud in the guise of a software application accessed through a client on the user's computing device (most often a web browser). Other than user preferences, consumers have no control over the software and the underlying computing resources that power it—for example, they must adapt to whatever changes (e.g., new or discontinued features) the software provider chooses to implement.[11]

The second category, typified by providers such as Amazon's Elastic Compute Cloud (EC2) or Rackspace, is primarily directed at system administrators who manage institutional computing resources. In this *Infrastructure as a Service (IaaS)* model, service providers provide access to computing resources in the guise of virtual images of traditional desktops machines or servers. Virtual machines are available in a broad range of capacity and pricing models—from pay-as-you-go micromachines to multi-year commitments on massively parallel systems, and even spot markets for bidding on cheap excess capacity. By allowing customers to reuse their installed software base, such a system allows them to enjoy the economic benefits of the Cloud with minimal disruption.

Beyond these two types of service access (software and virtual machines) lies an expansive *terra incognita*, currently referred to as *Platform as a Service (PaaS)*. Directed at software developers and exemplified by the Google App Engine and Amazon's AWS Elastic Beanstalk, this model offers direct access to the vast computing resources of the Cloud, so long as customers are willing to rewrite their software to take advantage of this new mode of provisioning software, processors, storage, and network resources. In effect, then, PaaS refers to the reinvention, in the context of the Cloud, of the software abstractions (e.g., files and process) that currently live within traditional desktop operating systems. New abstractions [e.g., the Google File System (Ghemawat, Gobioof, and Leung 2003) and MapReduce (Dean and Ghemawat 2008)] have already been developed by cloud providers for their own internal use and are being offered to the larger software community. There is much computational efficiency to be gained in using abstractions directly tailored to the computing resources of the Cloud (e.g., scalability), but, as in the desktop world, lack of standardization between competing platforms will pose significant hurdles for software developers.

In addition to these service delivery models, cloud resources can be deployed in distinct modes: *privately*, for consumption within institutions themselves; *publicly,* as a service to consumers; or as a pool of *community-owned* resources. In each case, resources can be deployed *on site*, *outsourced*, or various *hybrid* models, such as so-called *cloud-bursting*, where an institution manages its day-to-day workload internally but delegates peak demand to outside providers.

Tackling the Mystery

The Cloud, then, embodies a specific moment in the evolution of computing: the moment where, symbolically and materially, it becomes infrastructure and where a broad range of policy concerns become visible and open to debate. This moment is not static: there exists a wide range of current and possible future incarnations of the Cloud, with different implications for costs, control, reliability, interoperability, and of course, traditional public policy goals of equitable access, economic competitiveness, efficiency, etc. Indeed, focusing on the Cloud as infrastructure allows us to move beyond long-cherished mythologies of computing technologies as immaterial, infinitely expansive, and fundamentally democratic.[12] From such an infrastructural perspective, themes more useful for analytical work emerge.

First, the Cloud, like other transportation, energy, and communication infrastructures, is fundamentally driven by the economics of *capacity* and *demand*. As Joe Weinman makes clear in chapter 1 of this volume, "Cloud Strategy and Economics," the Cloud is first and foremost about a simple move: pay-per-use, the shift from ownership to rental of computing resources and their consumption at a much finer level of granularity than previously possible. Providers enjoy the economies and efficiencies that result from resource pooling and capacity planning, while consumers enjoy the ability to scale computing resources up and down as their needs require. Yet, as Weinman points out, for organizations to effectively capture value from the Cloud will require more than simply dumping their servers at the curb. Many additional factors will demand attention, including the specific shape of the demand, the spatial distribution of data centers and its impact on response time, and even, perhaps less intuitively, psychological factors, such as "flat-rate bias," that actively shape consumers' perception of the pros and cons of switching to the Cloud. While such concepts have long been deployed in the analysis of other infrastructures, they are only beginning to be used in the context of computing.[13]

Questions of capacity and demand immediately raise the complementary issues of *criticality* and *reliability*. In chapter 2, "Finding Security in the Cloud," Marjory S. Blumenthal explores the consequences of our far-reaching dependence on the Cloud and its de facto status as critical infrastructure, while in chapter 3, "Reliability and the Internet Cloud," William Lehr explores the conditions under which the Cloud might provide assurances of its reliability. These are particularly thorny issues: in contrast to the telecommunications industry, the computing industry and the Internet have developed largely outside regulatory purview. Furthermore, as the somewhat unified, vertically integrated market structure of the past has given way to the highly heterogeneous, vertically disintegrated structure of today, the technical and economic structure of the

Cloud has grown correspondingly complex. How can policy makers influence the behavior of a system whose properties depend on the interaction of millions of parts interconnected through modular interfaces? And how can regulation promote the public's interest in a reliable and secure infrastructure while simultaneously preserving the high rate of innovation that has characterized the computing industry to date?

The issue of how to best coordinate the evolution of a largely decentralized, modular infrastructure is also raised by Christopher S. Yoo in chapter 4, "Cloud Computing, Contractibility, and Network Architecture." As more types of computing resources are accessed through networks, new kinds of traffic (e.g., telesurgery) are requiring service guarantees that go beyond the best-effort principle that continues to characterize current terms from service providers—users might need, for example, to ascertain the identity and reliability of the specific paths through which their data circulates. Yet, as Yoo reports, the current layered decomposition of the Internet stack has proved extraordinarily difficult to adapt to its new circumstances. Yoo proposes that the imbrication, within a new layer, of legal primitives drawn from contract theory might provide the means to enforce service guarantees on Internet traffic. While we are only beginning to understand and address the challenges of gracefully evolving modular systems, Yoo's proposal demonstrates how multidisciplinary approaches that bridge the policy/engineering divide can be usefully brought to bear on such issues.

Yet another major concern is that of *privacy* and *liability*. The revelations by Edward Snowden in 2013 about the data collection practices of the US National Security Agency (NSA) have, among many other things, highlighted how the centralized nature of the Cloud has proved particularly convenient to the surveillance efforts of the US government. In chapter 5, "Cloud Privacy in the United States and the European Union," Andrea Renda observes the widely divergent legal and regulatory environment for privacy that obtains in the United States and the European Union in the post-Snowden era. In the first case, a somewhat haphazard patchwork of federal, state, and case law has largely focused on the risks of governmental intrusion; in the second case, the more conceptually coherent and centralized framework of a European directive has focused on the risks of private sector intrusion. In both cases, taxonomies of service providers used to allocate liability map only with difficulty on the rapidly evolving Cloud ecosystem.

Indeed, in chapter 6, "Understanding Regulatory and Consumer Interest in the Cloud," Jonathan Cave, Neil Robinson, Svitlana Kobzar, and

Helen Rebecca Schindler outline a broad range of consumer concerns that arise as part of the restructuring of various markets through the Cloud (e.g., code and data mobility, data breach, and content piracy). The take-it-or-leave-it approach of many providers' service-level agreements (SLAs), as well as the embedded and automated functionality of many Cloud services (Apple's iCloud being a prime example), conspire to produce an environment in which consumers experience many new and unfamiliar risks, often with minimal or confusing control over the parameters of their participation. The situation is likely to get much worse: the marriage of Big Data, Cloud services, and the interoperability made possible by the proliferation of application programming interfaces (APIs) promises to spawn an extraordinarily complex set of commercial relationships among data subjects, service providers, and data processors—making the task of regulators all the more challenging.

Another major theme can be articulated as that of *delocalization*: the Cloud is fundamentally about the movement of computing resources (whether processing, storage, software, or data) away from the desktop and towards geographically dispersed data centers. While this movement can be analyzed purely in terms of technical efficiency, it also has immediate implications in terms of *control*. Yet, as Luciana Duranti argues in chapter 7, "Digital Records and Archives in the Commercial Cloud," the trustworthiness of records has historically been grounded in the nature of the archive as a *physical place*. Given that public cloud providers typically offer no guarantees as to where specific data may reside, archivists must define new concepts, conditions, and terms of service that can continue to ensure proper chain of custody and the corresponding evidential value of electronic records.

To software manufacturers, on the other hand, the shift in control brought about by delocalization offers an extraordinary opportunity to resolve the long-standing problem of software distribution. Not only is the need to provide consumers with a physical copy of their software obviated by the Cloud, but the contractual relationship shifts from one of ceding ownership to one of providing a service. As Lothar Determann and David Nimmer eloquently demonstrate in chapter 8, "Software Copyright in the Cloud," legal scholars and practitioners have already struggled for years to reconcile fundamental concepts of copyright law with the specific material embodiment of software (including the inherent need for computers to make multiple working copies of the original software) and the thorny problem of distinguishing functionality from creative expression. The shift to the Cloud, however, might signify a turning point, and Determann and

Nimmer argue that its consequences might reach far beyond software to all downloadable content, including film, music, and literary works. Their chapter powerfully demonstrates how the computing infrastructure persists, not only in terms of physical wires and software abstractions, but also in terms of legal concepts carried over, for better or for worse, from otherwise forgotten episodes of technical history.

Finally, delocalization induces shifts not only in terms of control, but also in terms of *agency*. As Nicholas Bauch argues in chapter 9, "Bodies in the Cloud: A Geography of Electronic Health Data," laws and policies are increasingly recognizing that our cloud-based virtual representations operate in the world as functional extensions of ourselves, expanding our geographic presence beyond the confines of our mere biological bodies. Bauch coins the term *body-data* to recognize that electronic data, stored remotely in data centers and moving through digital networks, "contributes to the objecthood of something else (in this case, a human body) in a different location." The Cloud is Us, literally, and its geography, energy consumption, and waste are but an extension of our own.

On a historical scale, the gradual shift of computing from stand-alone tool to behind-the-scenes infrastructure has barely begun. It will take considerable time to achieve a level of ubiquity, integration, and invisibility on par with that of, say, the transportation infrastructure. As such, the chapters in this book are intended as starting points, to take stock of the situation and explore concepts and approaches that might prove useful in tackling this important transformation of computing. Obviously, many additional dimensions of cloud computing will benefit from further discussion and exploration. These include (to name just a few) standardization, as the inherently modular character of the Cloud puts increasing pressure on standardization as a market strategy and the governance of standardization bodies; power consumption, which, in the span of a few short years, has become the dominant design objective across all layers of computing, with implications for the public interest in green computing; and liability, as institutions as diverse as the financial sector, health care, and law enforcement come to depend on the Cloud for their day-to-day operations. As the Cloud becomes imbricated in the fabric of an ever-broadening array of individual and societal pursuits, the range of policy issues that it raises will continue to expand, with a corresponding need to continue and deepen the kind of interdisciplinary dialogue exemplified in the following chapters.

Notes

1. I capitalize the term *Cloud* to underline its symbolic quality, even as this symbol encompasses diverse and heterogeneous technical realities.
2. See Longstaff (2000) for a very useful meta-analysis of networked industries and their common characteristics from a regulatory standpoint, including bottlenecks, access, small versus large loads, and short versus long hauls.
3. On the concept of infrastructural dynamics, see Jackson et al. (2007), Bowker et al. (2010), and Sandvig (2013).
4. This should not come as a surprise since the signature technology of the Internet, packet switching, predominated over circuit switching precisely through the application of automated statistical analysis to its own traffic.
5. Cisco (2013) predicts that the sum of all forms of video (i.e., TV, video on demand, Internet, and peer-to-peer file sharing) will constitute approximately 86 percent of global consumer traffic by 2016.
6. See, for example, Bell, Hey, and Szalay (2009).
7. To cite one example: "The unprecedented evolution of computers since 1980 exhibits an essentially exponential speedup that spans 4 orders of magnitude in performance for the same (or lower) price. No other engineered system in human history has ever achieved that rate of improvement; … Whole fields of human endeavors have been transformed as computer system capability has ascended through various threshold performance values." (Millett and Fuller 2011, 25).
8. I have argued (Blanchette 2011, 2012) that a focus on infrastructural forces and the material foundation of computing resources provides an appropriate pedagogical framework for teaching the historical evolution of computing over the current emphasis on mathematical abstraction.
9. Ironically, in many cases, the transfer of data sets to the Cloud can be cost-prohibitive, leading Ambrust et al. (2009, 16) to recommend the "FedEx disk option" that is, shipping them using a more traditional infrastructure.
10. An excellent overview of these considerations is provided by Badger, Grance, Patt-Corner, and Voas (2012), which builds on the work of Mell and Grance (2011).
11. The increasingly blurry boundaries between personal devices and the cloud should also be noted. While this has been the case for some time with respect to content (such as music, photos, and streaming content), it is also increasingly the case with respect to software itself. Both Google and Apple provide for automatic updating of software so that applications, operating systems, and cloud services are optimally synchronized with one another. In addition, Amazon's Silk Browser, first deployed with its Kindle Fire, distributes its processing needs between the Kindle's processor and Amazon's cloud infrastructure (Thomas, Jurdak, and Atkinson 2012).
12. See, for example, the classic statements by Barlow (1996) and Negroponte (1996), and more recently, Abelson, Ledeen, and Lewis (2008) and Gleick (2011) on the digital as the evolutionary end point of information.

13. See, for example, Barroso and Hölzle (2009) on the engineering side, and on the policy side, chapter 13 of Frischmann (2012) for its application of these concepts to network neutrality.

References

Abelson, Hal, Ken Ledeen, and Harry Lewis. 2008. *Blown to Bits: Your Life, Liberty, and Happiness after the Digital Explosion*. Boston: Addison-Wesley Professional.

Armbrust, Michael, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, 2009. Above the Clouds: A Berkeley View of Cloud Computing. Electrical Engineering and Computer Sciences, University of California at Berkeley, Technical Report No. UCB/EECS-2009–28. Available at http://www.eecs .berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html.

Armbrust, Michael, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, (April 2010). A View of Cloud Computing. *Communications of the ACM* 53 (4): 50–58.

Badger, Lee, Tim Grance, Robert Patt-Corner, and Jeff Voas. 2012. Cloud Computing Synopsis and Recommendations. National Institute of Standards and Technology Special Publication 800–146. Available at http://www.nist.gov/customcf/get_pdf .cfm?pub_id=911075.

Baldwin, Carliss Young, and Kim B. Clark. 2000. *Design Rules: The Power of Modularity*. Vol. 1. Cambridge, MA: MIT Press.

Barlow, John Perry. 1996. A Declaration of Independence of Cyberspace. Available at https://projects.eff.org/~barlow/Declaration-Final.html.

Barroso, Luiz André, and Urs Hölzle. 2009. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. San Rafael, CA: Morgan and Claypool Publishers.

Bell, Gordon, Tony Hey, and Alex Szalay. 2009. Beyond the Data Deluge. *Science* 323 (5919): 1297–1298.

Blanchette, Jean-François. 2011. A Material History of Bits. *Journal of the American Society for Information Science and Technology* 62 (6): 1042–1057.

Blanchette, Jean-François. 2012. Computing as if Infrastructure Mattered. *Communications of the ACM* 55 (October): 32–34.

Bowker, Geoffrey, Karen Baker, Florence Millerand, and David Ribes. 2010. Towards Information Infrastructure Studies: Ways of Knowing in a Networked Environment. In *International Handbook of Internet Research*, ed. Jeremy Hunsinger, Lisbeth Klastrup, and Matthew Allen, 97–117. Berlin: Springer.

Cisco. 2013. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012–2017. Available at http://newsroom.cisco.com/documents/10157/1142732/Cisco_VNI_Mobile_Data_Traffic_Forecast_2012_2017_white_paper.pdf.

Clarke, Richard A., Michael J. Morell, Geoffrey R. Stone, Cass R. Sunstein, and Peter Swire. 2013. Liberty and Security in a Changing World. Report and Recommendations of the President's Review Group on Intelligence and Communications Technology. Available at http://www.whitehouse.gov/sites/default/files/docs/2013-12-12_rg_final_report.pdf.

Dean, Jeffrey, and Sanjay Ghemawat. 2008. MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM* 51 (1): 107–113.

Edwards, Paul N. 2002. Infrastructure and Modernity: Force, Time, and Social Organization in the History of Sociotechnical Systems. In *Modernity and Technology*, ed. Thomas J. Misa, Phillip Brey, and Andrew Feenberg, 185–225. Cambridge, MA: MIT Press.

Federal Communications Commission (FCC). 2010. The National Broadband Plan. Washington, DC. Available at http://transition.fcc.gov/national-broadband-plan/national-broadband-plan.pdf.

Frischmann, Brett. 2012. *Infrastructure: The Social Value of Shared Resources*. Oxford, UK: Oxford University Press.

Fuller, Matthew. 2008. *Software Studies: A Lexicon*. Cambridge, MA: MIT Press.

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. 2003. The Google File System. *Operating Systems Review* 37 (5): 29–43.

Gleick, James. 2011. *The Information: A History, A Theory, A Flood.* New York: Pantheon.

International Business Machines Corporation. 1953. Principles of Operation: Type 701 and Associated Equipment. New York. Available at http://bitsavers.trailing-edge.com/pdf/ibm/701/24-6042-1_701_PrincOps.pdf.

Jackson, Steve, Paul N. Edwards, Geoffrey Bowker, and Cory Knobel. 2007. Understanding Infrastructure: History, Heuristics, and Cyberinfrastructure Policy. *First Monday* 12 (6).

Lanier, Jaron. 2010. *You Are Not a Gadget*. New York: Random House.

Lessig, Lawrence. 2001. The Architecture of Innovation. *Duke Law Journal* 51: 1783.

Longstaff, P. H. 2000. Networked Industries: Patterns in Development, Operation, and Regulation. Program on Information Resources Policy, Center for Information Policy Research, Harvard University. Available at http://www.pirp.harvard.edu/pubs_pdf/longsta/longsta-p00-2.pdf.

Mell, Peter, and Tim Grance. 2011. The NIST Definition of Cloud Computing. National Institute of Standards and Technology, Special publication 800–145. Available at http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

Millett, Lynette I., and Samuel H. Fuller, eds. 2011. *The Future of Computing Performance: Game Over or Next Level?* Washington, DC: National Academies Press.

Mosco, Vincent. 2014. *To the Cloud: Big Data in a Turbulent World*. New York: Paradigm.

Negroponte, Nicholas. 1996. *Being Digital*. New York: Random House.

Roscoe, Timothy. 2006. "The End of Internet Architecture." In *Proceedings of the 5th Workshop on Hot Topics in Networks*. Available at http://conferences.sigcomm.org/hotnets/2006/roscoe06end.pdf.

Sandvig, Christian. 2013. The Internet as Infrastructure. In *The Oxford Handbook of Internet Studies*, ed. William H. Dutton, 86. Oxford, UK: Oxford University Press.

Thomas, Bryce, Raja Jurdak, and Ian Atkinson. 2012. SPDYing up the Web. *Communications of the ACM* 55 (December): 64–73.

Weinman, Joe. 2012. *Cloudonomics: The Business Value of Cloud Computing*. Hoboken, NJ: John Wiley & Sons.

Wu, Tim. 2011. *The Master Switch: The Rise and Fall of Information Empires*. New York: Random House.

Zittrain, Jonathan. 2008. *The Future of the Internet—and How to Stop It*. New Haven, CT: Yale University Press.