CA Press

How Clouds Hold IT Together

Integrating Architecture with Cloud Deployment

Marvin Waschke

technologies

HOW CLOUDS HOLD IT TOGETHER

INTEGRATING ARCHITECTURE WITH CLOUD DEPLOYMENT

Marvin Waschke

ca technologies

Apress®

How Clouds Hold IT Together: Integrating Architecture with Cloud Deployment

ISBN-13 (pbk): 978-1-4302-6166-7

ISBN-13 (electronic): 978-1-4302-6167-4

Managing Director: Welmoed Spahr
Acquisitions Editor: Robert Hutchinson
Technical Reviewer: Efraim Moscovich
Editorial Board: Steve Anglin, Mark Beckner, Louise Corrigan, James DeWolf, Jonathan Gennick, Robert Hutchinson, Celestin Suresh John, Michelle Lowman, James Markham, Susan McDermott, Matthew Moodie, Jeffrey Pepper, Douglas Pundick, Ben Renow-Clarke, Gwenan Spearing
Coordinating Editor: Rita Fernando
Copy Editors: Jana Weinstein, Kim Wimpsett
Compositor: SPi Global
Indexer: SPi Global
Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media, LLC., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springer.com.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com.

Dedicated to my friends and colleagues
at CA Technologies

Contents

About the Author

Marvin Waschke was a senior principal software architect at CA Technologies. His career has spanned the mainframe to the cloud. He has coded, designed, and managed the development of many systems, including accounting, cell tower management, enterprise service desks, configuration management, and network management.

He represented CA Technologies on the DMTF Cloud Management Working Group, DMTF Open Virtualization Format Working Group, DMTF Common Information Model REST Interface Working Group, OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) Technical Committee, DMTF Cloud Auditing Data Federation Working Group (observer), DMTF Configuration Database Federation Working Group, W3C Service Modeling Language Working Group, and OASIS OData Technical Committee (observer).

He was the editor-in-chief of the CA Technology Exchange (an online technical journal) and the author of *Cloud Standards: Agreements That Hold Together Clouds*.

About the Technical Reviewer

Efraim Moscovich is a software architect and software engineering leader.

He has more than 25 years of experience in IT and software engineering in various capacities.

His areas of expertise and interest include cloud computing, automation, large-scale software architecture, complex event processing, usability, and ideation and innovation skills.

He served as the senior principal architect in the Architecture group at the Office of the CTO and the senior director of engineering for the Server Automation product suite at CA Technologies.

Efraim was a member of the CA Council for Technical Excellence and of industry standards groups such as the OASIS TOSCA technical committee, and he served as the lead and mentor on the CA Patent Review Board.

Acknowledgments

I have learned a great deal about cloud architecture and deployment from my many colleagues on the various standards working groups in which I have participated. Notable among these are Larry Lamers of VMware, Shishir Pardikar of Citrix, Alan Sill of the Open Grid Forum, Jacques Durand of Fujitsu, Eric Wells of Hitachi, and Ali Ghazanfar of ZTE Corporation.

I would like to give recognition to Paul Lipton, my friend and CA Technologies representative on the DMTF board of directors. Paul is also co-chairman of the OASIS TOSCA Technical Committee. Paul is a longtime participant in many computing standards groups and has been a friend and colleague for several years.

Jacob Lamm of CA is the executive sponsor for this book. Jacob and I have worked together for many years on enterprise management solutions, including service desk and configuration management systems. I have learned a great deal from Jacob about the business side of IT.

Efraim Moskovitz served as technical reviewer for this book. Without his help, there would have been many more gaps and gaffes. Efraim and I worked together on the CA Council for Technical Excellence for several years, and Efraim contributed several articles to the CA Technology Exchange, which I edited. He is a good friend and valued advisor.

Karen Sleeth is the director of the CA Press who encouraged me many times when I was feeling down about this project. The book would not exist without her.

I also am grateful to the folks at Apress: Robert Hutchinson whose wit and insight shaped many aspects of the book; Rita Fernando, the project manager who gently and tolerantly nudged me forward; and Kim Wimpsett, whose patient copyediting pruned away the excess shrubbery and clarified my obscure and twisted prose.

Finally, I must gratefully acknowledge my wife, Rebecca; my daughter, Athena; and my grandsons, Matthew and Christopher. They tolerated my erratic and reclusive ways while I was working on this book. I cannot thank them enough for this.

Introduction

Cloud computing is changing the computing industry and our daily lives. Without clouds, the search engines like Google that have become an ever-present source of information on every subject would not be possible. There would be no social networking sites like Facebook. Online shopping sites like Amazon would be cramped in scope and unpredictable in performance. Scientific projects like the Human Genome Project would be much more difficult. Businesses are moving more of their IT to cloud implementations every day.

Centralizing data into a cloud that is accessible to mobile devices such as laptops, tablets, and smartphones, as well as desktops on the enterprise premises, has proven successful and has aided in expanding the boundaries of the workplace far beyond the walls of the enterprise office. The nature of the workplace and work styles have changed and forced enterprise governance to change also.

The basics of enterprise software design apply to all enterprise software, but cloud implementations bring new requirements and challenges.

Greater participation in planning by business management is one requirement. Clouds supplied by a cloud service provider rather than owned by the enterprise have distinct advantages, but the presence of a third party complicates the relationship between the enterprise and its IT systems. The involvement of a cloud provider can raise questions about governance, security, and auditability that require the participation of business management. Business factors such as costing methods and accounting for ownership are also likely to change.

Technical solutions may help, but they cannot resolve these business issues. Consequently, cloud projects require intense cooperation between business managers and technologists that similar projects confined to the enterprise premises do not require.

Cloud implementations also require new technical skills that stem from the greater role of network communications and the use of a virtual environment. These issues can be especially challenging when implementing enterprise integration. On the other hand, clouds also present important opportunities for rationalizing and increasing enterprise integration, and the central nature of cloud implementations simplifies and increases the opportunities for integration. On the other hand, the rise of mobile devices and wider geographic distribution of enterprises bring on new technical challenges.

Cloud implementations can be central to reducing the investment in enterprise computing while increasing its functionality and reach. It can also be the key to enterprise integration, but to fully realize the potential, enterprises must undertake cloud computing projects as both business and technical efforts. These projects require a planning, development, and operations methodology that brings these aspects together systematically. The IT Infrastructure Library (ITIL) is a proven set of practices for bringing together business management and technical expertise for the development and management of successful IT services. This book brings cloud computing and service management together with detailed recommendations for successful services built on cloud technology.

PART

I

Services, Virtualization, Handhelds, and Clouds

CHAPTER

1

The Imperative

The Challenge of the Information Age and IT

SUMMARY

There is debate among economists and skeptics from other disciplines on the significance of the Information Age. For the last fifty years, Moore's law has accurately predicted the exponential growth of computing capacity. If exponential growth continues, exceedingly rapid growth will occur at some point, but when the curve will become steep is hard to predict. IT has seen rapid growth in the first decade of the twenty-first century. The rate of change seems to be increasing and IT is penetrating into lives in ways that were impossible a few years ago. IT may have reached escape velocity. Service-orientation, cloud, and IT integration are critical in harnessing the blast of new capacity.

For many people today, *information technology* (IT) is troubling.[1] The benefits from computing have always been apparent. But from the beginning prophets have warned that humans might someday become dominated by their machines. Yesteryear's dystopic speculation has given way to today's icy realism. The stagnant unemployment rate and the soaring stock market following the 2008 financial crisis raise questions. Have we reached a point where businesses can increase output and profit without employing more people? Have the machines really begun to replace humans? Where does IT fit in?

[1]The term *IT* is used differently in different contexts. Among engineers, it usually refers to the combination of hardware and software that makes up networked computer systems. In service management, the term usually also includes the department or group that is charged with developing and maintaining these systems and includes people, processes, and governance. IT is often also used as a general term for computing disciplines. In the present work, *IT* is almost always used in the service management sense.

Rumination over the potential elimination of jobs by computers has been a popular topic for many intellectuals, from science fiction writers to economists and statesmen. The standard retort to the fear of job elimination has been that computer-driven increases in productivity go hand in hand with increases in demand that generate more, not fewer, jobs. This answer has been borne out through decades of prosperity. Has the balance changed? Did the triumph in 1997 of the IBM supercomputer Deep Blue over the reigning World Chess Champion, Garry Kasparov render grand masters obsolete? Did the defeat in 2011 of *Jeopardy!* champion Ken Jennings by Watson mark a turning point in the economy?

Information technology has evolved rapidly since its inception, and enterprise computing has changed with it. Cloud service management is an approach to designing and managing enterprise IT that represents much of the progress that has been made in computing, particularly the progress of the last decade and a half.

This book examines three related strands of recent IT development: enterprise integration, service management, and cloud. These three strands all have roots deep in the history of computing, but they are combining and blossoming, beginning to dominate both enterprise and consumer computing, and changing national economies. I will explain these aspects of IT architecture, how they influence business and society, and how to plan and design projects based upon them.

I am a software architect, development manager, and a developer; not a stock market analyst, a business executive, or a product manager, but this book is meant for all these roles. Stock market analysts want to know where the next big thing will drive the market; executives look for a map to guide their company to success; and product managers are on the prowl for attractive new products and features to offer to their customers.

Inevitably, the software architect in me will often show in this book. This is not to slight the other important participants. Architects are often in the background supporting visionary leaders who introduce new products and concepts to consumers. Architects also have vision, but it is of a different type. They envision the software and its interaction with physical or virtual hardware, its strengths and limitations, how it can be made reliable and efficient in production. They must also anticipate that successful software continually changes and adapts to evolving business and technical environments. The architect's view encompasses the entire IT system, the integration of all its components, both those deployed on a cloud and traditional components on the enterprise premises. The architect's down to earth concentration on the workable and feasible often offers insights critical to entire cloud projects.

Until outsourcing became popular, IT management assumed that every aspect of the datacenter was under direct control of the IT department. Cloud further reduces IT control. When a cloud is public, personnel outside the enterprise reporting structure have control of equipment, data and production processes that were traditionally entrusted to enterprise IT.

These changes affect everyone. Executives must institute new ways of assuring that systems deliver services that align with corporate goals and are accountable to enterprise governance. Product managers have a range of challenges and opportunities to transform into products for their customers. Developers have new technologies to master and new ways of working on projects that have unfamiliar requirements and new goals.

An architectural view of cloud straddles the visionaries who anticipate the final product, the executives and managers who guide IT, and the developers who design components and write the code. The architectural view provides both perspective and practical patterns and guidelines for planning and creating a successful cloud implementation in an environment that combines cloud and on-premises components. Visionaries can gain insight into what is feasible and what is not and increase the depth of their vision. Executives and managers will see aspects of clouds that will change the way they conduct business. Developers will get some well-deserved help in understanding the both new and perhaps strange requirements and technologies that cloud puts before them.

Advances in Computing

Computers have become more accessible and even indispensable to a wide swathe of the population in both developed and less-developed nations. Some of the new ways in which we interact with computers are obvious. Social networking applications such as Facebook have worked one kind of transformation. Social networking takes the near-universal access to networked computers now prevalent and adds content with universal personal appeal. This means that computers are used by segments of the population that would not have touched a computer a short time ago. For example, my octogenarian mother-in-law got her first computer only a few years ago. She now sends and receives email and posts photographs and comments for friends and family daily, but she would not have considered owning a computer in 2000.

When I took a chemistry class with a computing project at university, students were not allowed to touch a computer. We passed in punch cards and received printouts through what looked like a bullet-proof security window for a check-cashing service in a bad neighborhood. The irony is that the old IBM 7094 behind the security wall did not have a hundredth of the computing or storage capacity of my mother-in-law's laptop. The computing power, network capacity, and storage required for a single Facebook page were not available to a major research lab in 1967, when I took that class.

The success of phenomena such as Facebook depends on the technology that makes social networking possible, but it also depends on a society that has evolved to the point that both the technology is accessible and the content that people want is available. This requires computing and storage capacity that far exceeds the requirements of IT of the past.

The smartphone has worked a related transformation by changing where we compute. When we roll up our sleeves to do some computing today, we don't go to a datacenter. There are desktops in our offices; we tote laptops to and from work; and a fourteen-year-old packs more computing power to the beach in a smartphone than NASA could muster for the Apollo moon mission.

Networking technology, especially the Internet, has transformed the independent, isolated computers of the past developed by the great innovators such as Gates, Jobs, and Wozniak. Computers are now embedded in an enormous network that links millions of end-user and server computers together. If it were not for security walls, checkpoints, and other intentional barriers, naughty computer users today would have direct access to nearly every other computer on the planet for fiddling with their data and flashing annoying pictures on their screens.

The amount of storage available today is difficult to describe with the terms used when computers first became consumer products. When the PC was introduced, storage was commonly on floppy disks and measured in thousands of bytes. Internal hard drives were quickly added to standard PCs. Early PC hard drive capacities were measured in millions of bytes (megabytes), and then billions of bytes (gigabytes). Today, trillion-byte (terabyte) PC hard drives are common and data centers have much more storage. Amazon Web Services (AWS) S3 storage service describes their storage in "objects." These objects can contain up to five trillion bytes (five terabytes) of storage. In early 2012, AWS reported storing 762 billion objects.[2] Facebook is said to deploy seven quadrillion bytes (seven petabytes) of new storage a month.[3] To put these quantities of data in perspective, the books in the Library of Congress are estimated to contain less than ten petabytes of data.[4]

[2]Rich Miller, Data Center Knowledge, "Amazon: 762 Billion Objects Stored on S3 Cloud," http://www.datacenterknowledge.com/archives/2012/01/31/amazon-762-billion-objects-stored-on-s3-cloud/, January 31, 2012.
[3]Rich Miller, Data Center Knowledge, "Facebook Builds Exabyte Data Centers for Cold Storage," http://www.datacenterknowledge.com/archives/2013/01/18/facebook-builds-new-data-centers-for-cold-storage/, January 18, 2012.
[4]The estimate is imprecise. See Leslie Johnston, Library of Congress, The Signal: Digital Preservation, "A 'Library of Congress' Worth of Data: It's All in How You Define It," http://blogs.loc.gov/digitalpreservation/2012/04/a-library-of-congress-worth-of-data-its-all-in-how-you-define-it/, April, 2012.

The implications of *big data* have captured the public eye.[5] Big data is a response to the burgeoning scale of computing. We now have the capacity to store previously unimaginably vast quantities of data, the computing power to process it, and ways to collect data that were unheard of ten years ago. Big data is technology to cope with this explosion of resources and to marshal them to generate useful knowledge. Without storage capacity, computing power, or abundant data collections, big data would simply not exist. But with the confluence of these resources, we are able to extract buying patterns, predict trends, detect terrorist plots, and make other discoveries that were not possible before. This ability presents challenges to governance and privacy, but with the challenges, there are opportunities.

Businesses generate whole new lines of business and make others obsolete with computing innovations such as the MP3 file format, network bandwidth, and small players with large audio storage capacity that turned the recording industry upside down, or digital processing that has largely eliminated paper from the US payment system.[6] The military, prime mover of the early stages of computing, now must counter the threat of cyberwarfare no less assiduously than it does the threat of nuclear weapons.

Enterprise architects, designers, and planners face a call to action for the benefit of enterprises, individuals, and society. They must harness and extend the changes, expansion, and innovation in computing that stand before us.

Historical Background

The history of the development of the industrial, technical, and information society in which we live is a history of economic and social change driven by technical advances. Engineers may ask why this is important. Most software engineers I know are not out to reshape society. Instead, most of them want to participate in the construction of projects and products that serve a useful

[5]*Big data* has no crisp definition. It was coined in response to the increased size and changed nature of data that has begun to be collected and processed in the last few years. Generally, big data has three characteristics: it is large, measured in terabytes, petabytes, and larger units; to be useful, it must be processed rapidly; and it may not be in traditionally structured form.

[6]Use of paper checks by consumers has declined sharply, but the Federal Reserve estimates that they will survive for another decade. Behind the scenes, paper has almost completely disappeared. Prior to September 11, 2001, paper checks were hauled by truck and airplane from bank to bank for clearing. Aircraft grounding following 9/11 brought a near crisis in the financial system by delaying check clearing. Regulations were changed to allow electronic transfers of images. A decade later, almost all physical movement of paper checks has been eliminated. See David B. Humphrey and Robert Hunt, "Getting Rid of Paper: Savings from Check 21," working paper 12-12, Philadelphia: Federal Reserve Bank of Philadelphia, http://www.philadelphiafed.org/research-and-data/publications/working-papers/2012/wp12-12.pdf, May 2012.

purpose and find a ready market. Like everyone else, they want to earn a good living and to create something they are proud of. In seeking those goals, they make decisions daily, even hourly. With a better understanding of how the work fits into the overall pattern of society and how products have changed lives and perspectives, those decisions will, in small and large ways, be better and the overall results more satisfying for everyone.

The Ages

Economists and cultural commentators often call the age of computers and electronic data processing the *Information Age*. This age began in the second half of the twentieth century and continues today. IT integration, service management, and cloud computing are important developments in the progress of the Information Age.

Two preceding ages helped shape the world we live in today: the Industrial Age and the Technological Age. To place the Information Age in perspective—especially in economic perspective—you must know something about the preceding ages. It is especially important to identify the forces that have been deemed powerful enough by historians and economists to distinguish one age from another. These forces give us a clue as to what is most important in IT today.

The Industrial Age

The Industrial Age marks the beginning of the modern world that we all recognize: the world in which machines began to contribute substantially to human well-being. The age began in the late seventeenth century and extended into nineteenth century. It started with the invention of the steam engine. Early steam engines drove fans to ventilate coal mines in Britain. Dangerous gases and lack of ventilation limited mine depths and coal production. These steam engines, much less sophisticated than the engines of a few decades later, dispelled the gases and made deeper mines practical. The increased coal supply replaced wood as the dominant fuel source. The supply of wood was dwindling as Britain's forest had nearly disappeared at the end of the seventeenth century. The Industrial Age began.

The introduction of the steam engine delivered a one-two punch: first, a new way to transform raw energy to useful work; second, an abundant source of energy to fuel the transformation. Coal was the source and the steam engine was the transformer. Enormous changes began. With the power to transform raw materials rapidly into saleable goods, opportunities flourished. Tenant farmers migrated to factory towns where new industrial jobs, dismal by present standards, were far better than starvation on inefficient remote farms.

New means of transportation, such as steam trains, opened larger markets and furthered accelerated development. Agriculture increased in efficiency and a smaller rural population could provide food for a growing urban population.

The steam engine did not have a role in all these changes, but it was a catalyst that started the transformation. Macroeconomists point out that these changes represented major changes in the exchange of goods and services. Formerly, agriculture had produced scarcely enough food for the farmers themselves. Stimulated by new markets, farmers began to adopt new cultivation methods, and food supplies multiplied. This enabled the growth of cities and entire populations became consumers of rural agriculture.

Previously, the expense of transportation limited commerce to the exchange of rare and expensive commodities. Steam trains and ships decreased the cost of transportation and made the distribution of ordinary consumer goods practical. Before the appearance of industrial manufacturing, artisans exchanged their wares within small communities. In the Industrial Age, manufacturers produced the same wares centrally and used advanced transportation to distribute them cheaply to large markets.

These changes redistributed wealth. Before the Industrial Age, wealth was in the hands of aristocrats whose riches derived from hereditary privileges such as the power to tax and royal bequests. During the Industrial Age, industrial production and commerce became growing sources of wealth, and the relative wealth of the aristocracy diminished as the new wealth accumulated among the upstart classes.

The Technological Age

The next age, the Technological Age, also transformed markets, population distribution, and sources of wealth. The twentieth century was the age of technology. Some of the achievements of technology include advances in chemistry that brought explosives, herbicides, pesticides, synthetic fertilizers, plastics, pharmaceuticals, synthetic fibers, and a host of other substances. Technology also created the electrical power distribution grid, electric lights and appliances. Just as steam power characterizes the Industrial Age, electricity and petroleum-fueled internal combustion engines characterize the Technological Age. The current transportation system based on airplanes, diesel-powered trains, highways, automobiles, and trucks are all innovations of the Technological Age.

Technology reduced the number of agricultural workers needed to produce food for the rest of the population, causing further migrations to urban areas as small family farms consolidated into the equivalent of agricultural factories, furthering the restructure of the landscape.

Increasingly, automated electric machinery made factories more efficient and reduced the relative cost of ever-more-complex products. Electric appliances replaced much of the drudgery involved in simple housekeeping. This decreased the need for domestic servants, which were a large segment of urban populations in the industrial nineteenth century.

Efficient transportation transformed markets. Fresh fruit and vegetables shipped in refrigerated railroad cars and transport trucks changed diets as fresh produce became available in all seasons. Telephone, radio, and television transformed communications and homogenized culture.

Technology enabled the rise of an educated and skilled middle class prepared to perform increasingly sophisticated tasks in growing industries. Populations that had been concentrated in urban areas used their growing wages to move to suburban areas surrounding the cities.

The Information Age

From the macroeconomist's point of view, both the Technological Age and the Industrial Age generated new sources of wealth and new markets. Because the next age, the Information Age, is still in progress, we don't know how the age will end—but some economists and historians are not even sure that the Information Age qualifies as an economic age on the same order as the Industrial and Technological Ages.

Why do some economists and historians deprecate the significance of the Information Age? They maintain that, although much of the progress in computing has brought about improvements, these improvements have not dramatically changed the structure of society and the economy.

In their view, although computing has made agriculture more efficient, we have observed only an incremental continuation of the trend toward more efficient cultivation and distribution, which has reduced the number of farmers and increased overall output, but not engendered new directions in farming. Just-in-time inventory depends on computing and has made manufacturing more responsive and more efficient, but manufacturers still create the same types of products without major changes to the distribution system. These contrarians argue that the Information Age has brought important improvements but not dramatic change. In their view, the Information Age has had little impact on sources of wealth and distribution of markets and therefore does not qualify as an economic age at all. This is an assertion that bears examination.

What we call the Information Age began when computers emerged from wartime laboratories and began to influence the conduct of business in the mid-twentieth century. At first, the main focus of computing was military and scientific. Computers cracked codes, calculated complex weapon trajectories, and solved complex equations for weapons development. Military and scientific computing will always be important,[7] but the application of computing that soon gained the most attention was business.

From midcentury on, computers became more and more prevalent in business. At first, computers primarily automated back-office functions—the activities that customers only indirectly participate in, such as maintaining accounts, billing and accounts receivable, and tracking inventory. In contrast, customers interact directly with front-office functions, such as online stores.

Mainframe

Mainframes are direct descendants of the computers designed for military and scientific purposes. Early computers were designed to solve a single problem. If confronted with a different problem, the computer had to be rebuilt. Since rebuilding was expensive and troublesome, the earliest electronic computers were only applied to the most critical problems.

Of the many innovations that drove the beginning of the Information Age, arguably the seminal innovation was the one proposed by John von Neumann. In early single-purpose computers, the instructions for solving a problem were in the wiring of the computer. The instructions were changed by rewiring the logic controls. The von Neumann design converted the control instructions into input that was treated like the data for a specific problem.[8] This meant that a single computer could solve different problems by consuming a different set of control instructions instead of rewiring. Von Neumann's EDVAC was the first design for a programmable general-purpose electronic computer.[9] All modern computers, from mainframes to handhelds, even embedded control chips, have roots in von Neumann's design.

[7]Military computing has taken a new twist recently as cyberwarfare has transformed computing from a tool for developing weapons into both a formidable weapon in itself and a valuable and ubiquitous asset that must be protected. This transformation is important in assessing the growing significance of the information age and IT integration.

[8]Von Neumann's original description of a programmable computer appeared as "First Draft of a Report on the EDVAC" was published in 1945. The document was published prematurely and the concept inadvertently became unpatentable. A reproduction of the document can be found at `http://virtualtravelog.net.s115267.gridserver.com/wp/wp-content/media/2003-08-TheFirstDraft.pdf`

[9]The priority claims of EDVAC (electronic discrete variable automatic computer) and the slightly earlier ENIAC (electrical numerical integrator and computer) are controversial. The ENIAC was programmable in the sense that its modular design permitted reprograming by physical rearranging jumpers and cables. The EDVAC could be reprogrammed without physically changing the hardware. The EDVAC was designed by the same team that designed the ENIAC and construction of the EDVAC began before the ENIAC was complete. At the core of the controversy, von Neumann was a latecomer to the team. Although only von Neumann's name appears on the EDVAC design paper, other members of the team assert that the EDVAC concepts were developed prior to von Neumann's participation in the team.

Automating the back office was, and still is, the main thrust of mainframe computing. In the early days of business computing, the day's transactions would be collected—often via messengers collecting decks of punched cards—and then run through a batch program to update the accounts at the end of the day. The result was a new deck of punch cards representing the state of the accounting system at the end of the day. This system was not far from that of the nineteenth century by which clerks were condemned to balance the books to the last penny before leaving the counting house. Before direct-access persistent storage (typically disk drives) became widely available, the updated books were stored on punch cards or magnetic tape and a printed master listing of all the data in a long report. When someone needed data on an account, he looked it up in the printed report, which was not that different from a merchant in 1850 opening a handwritten journal or a ledger to check on the status of an account. At this stage, the computer had made accounting faster and more accurate, but business was still conducted in the same manner—a good example of computing improving and accelerating an existing process, leading to incremental growth, but not a transformation.

I have been describing a serial computing environment in which jobs run one at a time in sequence. In this environment, integration meant designing output that the next program could read as input. If an accountant wanted to compare data from two departments, she had to go to the paper reports for the two departments and look up the data, or someone had to write a program that would take the output for two departments and combine them to produce an answer. This corresponds to our nineteenth-century merchant going to two sets of journals and ledgers for two divisions of his business. Integration had not progressed much beyond the level in manual systems, although updating accounts had become much faster and more accurate. Again, incremental growth.

Batch programs are less often written now. Programs that segment activities into many independent transactions that are executed when the transactions occur are much more common. In a *transactional system*, as soon as the customer makes a purchase, the system processes the transaction and balances the books as the transaction occurs rather than waiting until the end of the day for a batch update. Thus, in a transactional accounting system, account balances are always available and up-to-date.

Instant availability of account information presents opportunities. Not only do the traditional back-office functions run much faster but the state of the system is always, at least in theory, available. Using a transactional accounting system, executives can check the balance of any account at any time and expect up-to-the-minute information instead of waiting until a batch job completes. Transactional processing meant that businesses, especially large businesses, could make informed decisions anytime without waiting for batch jobs to complete.

It took time for transactional processing to expand and become more sophisticated. In the early days of mainframes, software and its source code were bundled with hardware. Mainframe owners were free to modify the programs to fit their needs. Often the modified code was donated back to the hardware vendor for sharing with other customers, somewhat similar to open-source code today. The foundations of many of the early transactional processing systems were developed in this way. This included the IBM *Information Management Transaction Manager* (IMS TM) and the IBM *Customer Information Control System* (CICS).

Instant availability of the accounting information and other information such as inventory levels had wide implications and potentials that far exceeded the capabilities of the computing infrastructure of the time.

Transactional systems with their up-to-the-minute information had much more potential than early systems could support. Continuous maintenance and accessibility of the system state adds another level of sophistication and complexity to information processing, because systems can share state. This is powerful. In 1966, Douglas Parkhill anticipated that integrated systems would combine information from every aspect of a business into a single model.[10] This model provides current information in near real time and is also capable of simulating and projecting future states for planning, enabling businesses to adjust their investments to optimize for the future.

Businesspeople have always done this kind of planning, but they had to rely on estimates based on experience and intelligent guesswork without the aid of current, comprehensive, and accurate information. In addition to affording predictive value, comprehensive and accurate data spurs innovation and inspires businesses to modify processes and offer new services that could not be supported by earlier technology.

Timesharing

Until the advent of distributed systems and desktops, computers were expensive—often too expensive for a smaller business to purchase and providing more capacity than the business could use. Nevertheless these businesses had processes ripe for automation. Timesharing was a solution to this need.

Operating systems were designed to support more than one user at a time. This was done by doling out short intervals of processing time to each user and switching fast enough and separating data well enough to give the users the illusion that each was in complete control and the only one on the system.

[10]Douglas F. Parkhill, *The Challenge of the Computer Utility* (Reading, MA: Addison-Wesley, 1966).

Users were usually connected to the timeshared central system by leased lines: private connections from their key board and display to the computer. If the line were fast enough and the central computer were sufficiently powerful for the number of users, it would appear as if a timeshare computer miles away were in the same room.

Timeshare computing made computing available to many businesses that could not afford their own computer, and it also began the spread of knowledge of computing outside the largest companies and institutions, but it was still expensive and not widely available to individual consumers.

Distributed Computing

When distributed computing began to take the place of mainframes in the public eye, mainframe back-office computing continued to be important, but distributed computing began to move computing capacity from sacrosanct glass-house datacenters and into the hands of ordinary office workers.

Distributed computing adds both value and complexity to computing. A distributed computing system is made up of relatively small networked computers instead of a single central computer. In distributed systems, large numbers of small and powerful computers perform computing tasks. The small computers rely upon networking technology to transfer data and control between them. There are advantages to both distributed and centralized mainframe approaches. Although the distributed approach is now prevalent, there are many productive mainframe systems active and not likely to be replaced soon.

Both mainframes and distributed systems are dependent on networks, but in different ways. Mainframe networks connect the mainframe in a sort of hub-and-spoke pattern (Figure 1-1). The primary network data path is between the central computer and an end-user terminal or other peripheral such as a storage unit or printer. Today, the scenario is more complicated because mainframes usually intermingle with distributed systems and mainframes are linked together. A mainframe user terminal is often a workstation in a distributed system and the mainframe itself may also be a node in a distributed system, but the primary data path remains between peripherals and the central processor.

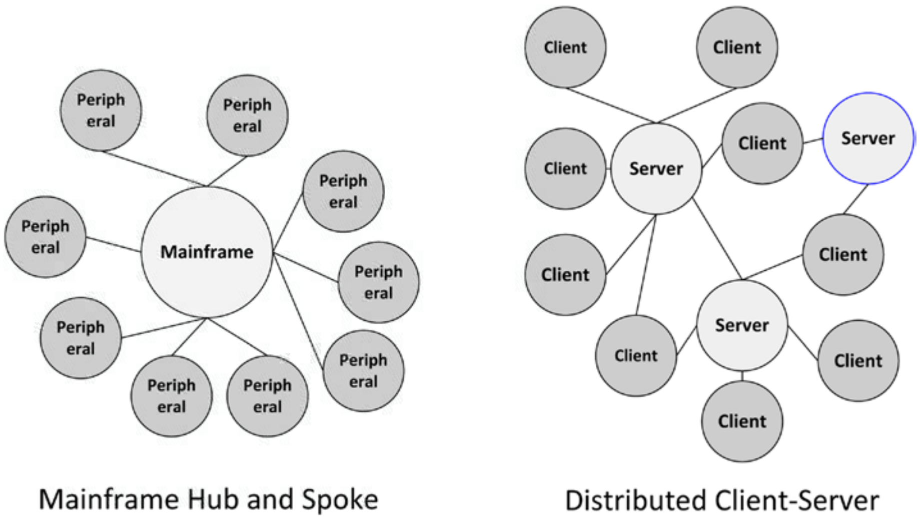

Figure 1-1. Mainframe and distributed systems have different characteristic network patterns

The components of a distributed system are often simpler and less sophisticated than the components of a mainframe system, but distributed systems as a whole can be more complex than mainframes. The network in a distributed system does not have a central computer. Instead, a distributed system is usually described as a client–server system. A client requests operations and data from a server. A server executes client requests and returns data and status to the client. A computer in a distributed system may switch around and act as a server in one transaction and as a client in the next. Often, to fulfill a client request, a server reaches out as a client to another server and acts as a client and a server at the same time.

Distributed computers are independent of the others in the system. If one fails, the others continue to operate, although the total system may be impaired because the software running on one computer may be dependent on the performance of software running on other computers.

Part of the challenge in building distributed systems is to prevent isolated failures from affecting the entire system. Failures in distributed systems can be complex. A hardware deficiency in one computer—a fragmented disk drive that delivers data more slowly than expected for example—may cause the entire system to perform more slowly, but the source of the slowdown may not be apparent and may require extensive troubleshooting.

Distributed systems also have dependencies on the network that mainframe systems do not have. Mainframe systems are usually dependent on a network for connecting the users to a central computer, but all computing occurs on the central computer with relatively little dependence on a network. In a distributed system, the same connection from the user to the system exists, but because servers interact with other servers and clients and servers can interchange roles, there is an added dependency on a connection between the computers that are performing the distributed computing work. A request from a client often will be parceled out to a number of servers and it may not be easy to determine where a problem has occurred. These issues can become even more complex when systems are virtualized, in which case computing may appear to be centralized but in fact be distributed over many networked physical machines.

Another important characteristic of distributed systems is scalability. A scalable system can be expanded to handle greater loads, often characterized by greater numbers of clients; it can also be shrunken when the load decreases. Often scalability is equated with supporting vast numbers of clients and huge loads. This is not exactly correct: a system designed to handle a specific load, no matter how large, is not scalable if it is not designed to be "scaled up" to even larger loads. "Scaling down" usually does not get the attention that scaling up receives, but under some circumstances, decreasing the capacity of the system can also be important.

Although mainframes handle very large loads—often larger than comparable distributed systems—distributed systems typically scale more easily; when the load is too large, additional servers are added to the system. If the system is designed well, there are few limits on the number of servers that can be added. Scaling up a single central computer can be more difficult because there is only the central computer to expand and there is a limit to the size and number of processors that can be added.

Computer theorists of the 1960s, such as Douglas Parkhill, did not anticipate the appearance of distributed systems. As discussed, distributed systems rely on compact powerful computers that are strung together into a system. Without the powerful desktops that began to appear in the 1980s, the current common architecture of distributed systems connected by the Internet would look far different. In hindsight, small and powerful computers were inevitable. As long as the processing power packed into a single electronic chip follows Moore's law and increases exponentially, as it has since the invention of the transistor, computers will become smaller and more powerful with each year.

This means, on the one hand, smaller and more powerful handhelds that can interpret speech more accurately and play movies flawlessly and, on the other hand, giant datacenters stuffed with hundreds of thousands of the same powerful processors closely linked together to perform monumental tasks such as searching the Internet or managing a large and growing social network.

Distributed systems have affected business computing in many ways. Desktop computers are cheap—very cheap, compared to mainframes. Purchasing or expanding a mainframe usually requires corporate budgeting, because mainframes are expensive. Desktop computers can be purchased under departmental or even individual budgets and often crept into enterprises with little strategic input from upper management. They proved to be so useful that they became a runaway success, often to the consternation of the professionals in the datacenter.

Soon every office desk had a desktop PC, and homes were not far behind. This presented an opportunity for software programmers and vendors who unleashed a flood of useful (and occasionally not so useful) applications for individual and departmental groups. Cheap desktops and equally cheap and innovative software rapidly spread through enterprises and began to affect productivity and business methods.

The proliferation of desktop distributed systems had several important implications. Activities were automated that had never been automated before. Word processing together with electronic filing is one example. Automated staff scheduling is another. With increased automation, productivity increased, and the amount of data that could be analyzed electronically also increased. The flaw in all this growth was lack of coordination and strategic direction. Artificial barriers to communication, such as departments with incompatible systems to manage the same task, proliferated. Although the data available for analysis grew rapidly, much of this data remained inaccessible. Data that could be used by the entire enterprise might be stored on a hard drive on an administrator's desk and protected and backed up at the whim of an untrained user.

Although independent personal applications, such as office productivity tools, are still important, management of distributed systems has evolved. Many applications that were once individual or departmental have been taken over by the enterprise and managed and maintained by the corporate IT department. Even personal applications such as word processors and spreadsheet tools are usually managed by corporate IT so they can be supported efficiently for security and reliability.

Distributed System Integration

Bringing diverse processes and data together is called *integration*. As IT systems become more integrated, the enterprise becomes more efficient and agile. Increased integration is at the heart of many of the advances in IT that we are so familiar with.

Although distributed systems have come to dominate IT, they present an integration challenge. The benefits from integration of distributed systems are the same as in mainframe systems. However, distributed data can be more difficult to bring together and analyze. Communication between distributed processes is often difficult. Integration of distributed systems is dependent on innovative use of the network. Many of the advances in system architecture of the past fifteen years are related to communication between distributed processes.

Impact of the Information Age

As mentioned, some economists maintain that the Information Age does not qualify as an age because it has not had social and economic consequences comparable to preceding ages. Looking at some of the characteristics of IT today is more revealing than arguing one side or the other. It may be that the changes the economists are looking for are not here quite yet but loom on the horizon.

Many advances in the last decade are obvious. Many people go for days without touching paper money, coins, or a checkbook. Credit and debit cards and online payment systems take care of it all. You can buy almost everything you need to live over the Internet today. You could probably go for months without entering a store, except perhaps a grocery store for fresh produce, although even fresh grocery shopping services are available in some cities.

It is possible that you have stopped reading paper newspapers and magazines, choosing to view them instead on laptops and tablets. You probably buy books online, and a substantial number of those are probably electronic, not paper. If you are a budding author, you may eschew traditional publishers to publish and sell your book online for only a few dollars. Many musicians record and self-publish their music electronically, circumventing the once all-powerful record companies.

Banking and insurance are two examples of industries that have been transformed and made more efficient by IT, but the industries themselves have not disappeared. Banks are able to transfer funds and track accounts much more efficiently now than they did fifty years ago, but they still provide services that are similar to those they offered in 1960. I am old enough to remember when banks closed their doors at 3 PM to balance their accounts, a check took weeks to clear if you did not take it to the bank that issued it, and everyone took hours each month to reconcile paper cancelled checks with the bank's accounting. Debit cards and online bill payment are much more convenient and the accounting is almost instantaneous, yet the services—relieving me of the risk of carrying large amounts cash and of the necessity of visiting payees in person in order to make payments—are the same services provided by bankers since the Middle Ages.

These kinds of instrumental and incremental changes are less radical in nature than the changes brought about by the Industrial Revolution, which transformed what people ate and where they lived, or those of the Technological Revolution, which gave rise to an entire educated middle class.

Nevertheless, the potential for IT to drive societal and economic changes is staggering. Parkhill predicted in 1966 a significant list of the features of the Information Age today: computerized shopping, the declining use of physical money, universal computerized access to most of the planet's information, and automated paperless publishing. But he also predicted high rates of unemployment and social change as the human role in the economy declines and is replaced by computing. If this prediction is realized, the Information Age will achieve full recognition as an economic age.

Parkhill saw the decline in employment as a benefit, not a detriment. Although he expected a period of transition that might be difficult, he predicted that eventually society would adjust and employment would be directed toward humanistic goals. Whether this prediction will prove as prescient as his others remains to be seen.

Moore's Law

The Information Age is essentially different from the other ages owing to *Moore's law*. Moore's law is not a law in the formal sense; rather, it is an inductive prediction that has proven correct for over fifty years, since the invention of the transistor in 1958. Moore predicted that the density of transistors on integrated circuits would double every two years. The impending failure of Moore's law has been predicted many times, but so far it has held up with remarkable consistency and the end is not in sight. Entirely new approaches to computing, such as quantum computers, portend a whole new realm of expansion beyond the limitations of silicon technology.

Moore's law projects exponential growth. Most people know what exponential growth means, or have at least heard the term, but the implications are often surprising. Let's examine exponential growth more carefully.

Exponential Growth

There are many stories of clever mathematicians using their knowledge of exponential series to trick the unknowing. One goes like this: a poor mathematician does a service to the king and the king offers to pay. The mathematician offers the king two payment plans: an arithmetic plan and an exponential plan (Figure 1-2). The first plan is to pay the mathematician one dollar today, two dollars tomorrow, three dollars the day after, increasing the payment by a dollar each day for thirty days. The second plan is to pay the mathematician half a cent today, one cent tomorrow, two cents the day after, and four cents the day after, doubling the amount each day for only thirty days.

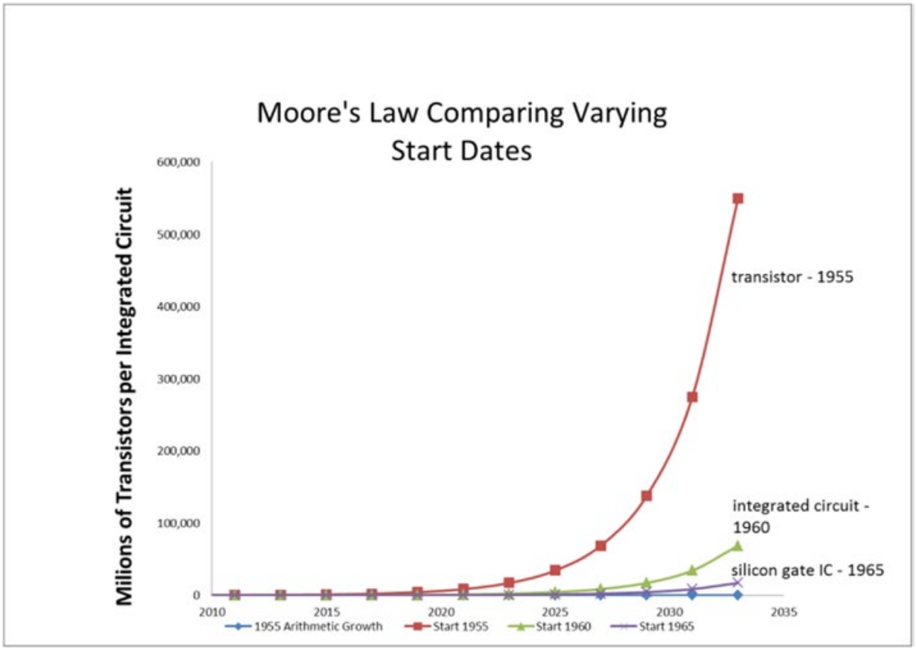

Figure 1-2. Changing the start date of Moore's law from 1955 to 1965 postpones a radical increase in computing power

The king chooses the second plan. We all know the outcome. If the king had chosen the first plan, he would have paid a total of $465. But by choosing the second plan, he will end up paying a total of $5,368,709.

We also know the real outcome. By the end of the third week when the payout to the mathematician exceeds $10,000, the king has the mathematician's head removed to prevent further disruption.

But there is a very important point in this story. Halfway through the month, on the fourteenth day, the first plan's total payout would have been $105, whereas the second plan's payout was $82. In other words, for the first two weeks, the exponential plan favored the king.

If the mathematician had made the exponential plan more attractive and reduced the first payment to five thousandths of a cent (0.005¢), the total payout would still be over half a million dollars, but the first plan would favor the king until the eighteenth day, and the payout would not exceed $10,000 until the twenty-fourth day. The mathematician would have lived an extra week.

By adjusting the starting point of the exponential payment plan, we can adjust the mathematician's life span. If we reduce the initial payment to 0.000005¢, he might live to old age because the total payout over thirty days is less than $6,000.

The Industrial and Technological ages were analogous to the first payment plan. The first steam engine was built, then the second, then the third, and so on. Each engine became more powerful; building the second steam engine doubled steam power, but building the third only added a third more power. Each additional engine added something to the previous, but there was not continuous doubling of power. The inventions of the Technological Age were similar: innovations were continuously added but there was not exponential growth. Society was certainly transformed by the innovations of each of these ages, but the transformations of these ages proceeded through incremental processes rather than exponential series.

The differences between incremental and exponential series are important. The rate of change in an incremental series stays constant. In the real world, increments differ in size, so the rate of change varies, but over the long haul, progress follows a straight line. At least in retrospect, incremental progress is easy to understand and projections are relatively simple and reliable.

Exponential series are less intuitively stable. To interpret an exponential series, you have to know where you are in the series. This tells us something important about the growth of the Information Age. If we accept Moore's law and peg the growth of the Information Age to the exponential growth of computing power that the law implies, we don't know if the starting point in 1958 was a dollar or a hundred-thousandth of a cent. The year 2014 might be analogous to the third week of a sequence that started at half a cent, in which case the Information Age is about to become exciting. But the sequence might have started at 0.005¢, in which case we have another couple decades to wait before the real excitement starts. We know the series is exponential, but we don't know where we are on the curve (see Figure 1-2). Moore's law does not tell us whether we have already entered the steep part of the curve, in which case the Information Age is a wash-out, or whether the curve is about to take off and we will soon find out what the Information Age is about, or we have decades to wait before the acceleration takes off. Figure 1-2 shows that if the Moore's law progression started in 1955 (the start date usually ascribed to the law), chip density will begin to take off in 2010 to 2020. If we have decades to wait, maybe Moore's law will finally fail and the Information Age will wash out in another way.