CHAPTER 1

Introduction

The ARPANET is nearly 50 years old, and the World Wide Web is close to 3 decades old. Yet the internet technologies that emerged from these two remarkable milestones continue to transform industries and cultures today and show no signs of slowing down. The widespread use of such popular internet services as web-based email, search, social networks, online maps, and video streaming, plus the increased worldwide availability of high-speed connectivity, have accelerated a trend toward server-side or "cloud" computing. With such trends now embraced by mainline enterprise workloads, the cloud computing market is projected to reach close to half a trillion dollars in the next few years [Col17].

In the last few decades, computing and storage have moved from PC-like clients to smaller, often mobile, devices, combined with large internet services. While early internet services were mostly informational, today, many web applications offer services that previously resided in the client, including email, photo and video storage, and office applications. Increasingly, traditional enterprise workloads are also shifting to cloud computing. This shift is driven not only by the need for user experience improvements, such as ease of management (no configuration or backups needed) and ubiquity of access, but also by the advantages it provides to vendors. Specifically, software as a service allows faster application development because it is simpler for software vendors to make changes and improvements. Instead of updating many millions of clients (with a myriad of peculiar hardware and software configurations), vendors need to only coordinate improvements and fixes inside their data centers and can restrict their hardware deployment to a few well-tested configurations. Similarly, server-side computing allows for faster introduction of new hardware innovations, such as accelerators that can be encapsulated under well-defined interfaces and APIs. Moreover, data center economics allow many application services to run at a low cost per user. For example, servers may be shared among thousands of active users (and many more inactive ones), resulting in better utilization. Similarly, the computation and storage itself may become cheaper in a shared service (for example, an email attachment received by multiple users can be stored once rather than many times, or a video can be converted to a client format once and streamed to multiple devices). Finally, servers and storage in a data center can be easier to manage than the desktop or laptop equivalent because they are under control of a single, knowledgeable entity. Security, in particular, is an important differentiator for the cloud.

Some workloads require so much computing capability that they are a more natural fit for a massive computing infrastructure than for client-side computing. Search services (web, images, and so on) are a prime example of this class of workloads, but applications such as language translation (and more broadly, machine learning) can also run more effectively on large shared computing installations because of their reliance on massive-scale models.

These trends toward server-side computing and widespread internet services created a new class of computing systems that, in our first edition of this book, we named *warehouse-scale computers*, or WSCs. The name is meant to call attention to the most distinguishing feature of these machines: the *massive scale* of their software infrastructure, data repositories, and hardware platform. This perspective is a departure from a view of the computing problem that implicitly assumes a model where one program runs in a single machine. In warehouse-scale computing, the program is an internet service, which may consist of tens or more individual programs that interact to implement complex end-user services such as email, search, or maps. These programs might be implemented and maintained by different teams of engineers, perhaps even across organizational, geographic, and company boundaries (as is the case with mashups, for example).

The computing platform required to run such large-scale services bears little resemblance to a pizza-box server or even the refrigerator-sized, high-end multiprocessors that reigned in prior decades. The hardware for such a platform consists of thousands of individual computing nodes with their corresponding networking and storage subsystems, power distribution and conditioning equipment, and extensive cooling systems. The enclosure for these systems is in fact a building, which is often indistinguishable from a large warehouse.

1.1 WAREHOUSE-SCALE COMPUTERS

Had scale been the only distinguishing feature of these systems, we might simply refer to them as *data centers*. Data centers are buildings where multiple servers and communication gear are co-located because of their common environmental requirements and physical security needs, and for ease of maintenance. In that sense, a WSC is a type of data center. Traditional data centers, however, typically host a large number of relatively small- or medium-sized applications, each running on a dedicated hardware infrastructure that is de-coupled and protected from other systems in the same facility. Those data centers host hardware and software for multiple organizational units or even different companies. Different computing systems within such a data center often have little in common in terms of hardware, software, or maintenance infrastructure, and tend not to communicate with each other at all.

WSCs currently power the services offered by companies such as Google, Amazon, Facebook, Microsoft, Alibaba, Tencent, Baidu, and others. They differ significantly from traditional data centers: they belong to a single organization, use a relatively homogeneous hardware and system software platform, and share a common systems management layer. Often, much of the application, middleware, and system software is built in-house compared to the predominance of third-party software running in conventional data centers. Most importantly, WSCs run a smaller number of very large applications (or internet services), and the common resource management infrastructure allows significant deployment flexibility. The requirements of homogeneity, single-organization control, and enhanced focus on cost efficiency motivate designers to take new approaches in constructing and operating these systems.

Although initially designed for online data-intensive web workloads, WSCs also now power public clouds such as those from Amazon, Google, and Microsoft. Such public clouds do run many small applications, like a regular data center. However, as seen by the provider, all of these applications are identical VMs, and they access large, common services for block or database storage, load balancing, and so on, fitting very well with the WSC model. Over the years, WSCs have also adapted to incorporate industry standards and more general purpose designs that the same infrastructure can support both large online internal services as well as large public cloud offerings, with very little deviation, and in many cases, even the same developer experience.

Internet services must achieve high availability, typically aiming for at least 99.99% uptime ("four nines," or about an hour of downtime per year). Achieving fault-free operation on a large collection of hardware and system software is hard and is made more difficult by the large number of servers involved. Although it might be theoretically possible to prevent hardware failures in a collection of 10,000 servers, it would surely be very expensive. Consequently, WSC workloads must be designed to gracefully tolerate large numbers of component faults with little or no impact on service level performance and availability.

1.2 COST EFFICIENCY AT SCALE

Building and operating a large computing platform is expensive, and the quality of a service it provides may depend on the aggregate processing and storage capacity available, further driving costs up and requiring a focus on cost efficiency. For example, in information retrieval systems such as web search, three main factors drive the growth of computing needs.

- Increased service popularity translates into higher request loads.
- The size of the problem keeps growing—the web is growing by millions of pages per day, which increases the cost of building and serving a web index.
- Even if the throughput and data repository could be held constant, the competitive nature of this market continuously drives innovations to improve the quality of results retrieved and the frequency with which the index is updated. Although some quality improvements can be achieved by smarter algorithms alone, most substantial improvements demand additional computing resources for every request. For example, in a search system that also considers synonyms of the search terms in a query, or semantic relationships, retrieving results is substantially more expensive. Either the search needs

to retrieve documents that match a more complex query that includes the synonyms, or the synonyms of a term need to be replicated in the index data structure for each term.

This relentless demand for more computing capabilities makes cost efficiency a primary metric of interest in the design of WSCs. Cost efficiency must be defined broadly to account for all the significant components of cost, including hosting-facility capital and operational expenses (which include power provisioning and energy costs), hardware, software, management personnel, and repairs. Chapter 6 discusses these issues in more detail.

1.3 NOT JUST A COLLECTION OF SERVERS

Our central point is that the data centers powering many of today's successful internet services are no longer simply a collection of miscellaneous machines co-located in a facility and wired up together. The software running on these systems, such as Gmail or web search services, executes at a scale far beyond a single machine or a single rack: it runs on no smaller a unit than clusters of hundreds to thousands of individual servers. Therefore, the machine, the computer, is itself this large cluster or aggregation of servers and needs to be considered as a single computing unit.

The technical challenges of designing WSCs are no less worthy of the expertise of computer systems architects than any other class of machines. First, they are a new class of large-scale machines driven by a new and rapidly evolving set of workloads. Their size alone makes them difficult to experiment with or simulate efficiently; therefore, system designers must develop new techniques to guide design decisions. In addition, fault behavior, security, and power and energy considerations have a more significant impact in the design of WSCs, perhaps more so than in other smaller scale computing platforms. Finally, WSCs have an additional layer of complexity beyond systems consisting of individual servers or small groups of servers; WSCs introduce a significant new challenge to programmer productivity, a challenge even greater than programming the individual multicore systems that comprise the WSC. This additional complexity arises indirectly from virtualization and the larger scale of the application domain and manifests itself as a deeper and less homogeneous storage hierarchy (Chapter 4), higher fault rates (Chapter 7), higher performance variability (Chapter 2), and greater emphasis on microsecond latency tolerance (Chapter 8).

The objectives of this book are to introduce this new design space, describe some of the requirements and characteristics of WSCs, highlight some of the important challenges unique to this space, and share some of our experience designing, programming, and operating them within Google. We are fortunate to be not only designers of WSCs but also customers and programmers of the platform, which has provided us an unusual opportunity to evaluate design decisions throughout the lifetime of a product. We hope that we succeed in relaying our enthusiasm for this area as an exciting new target worthy of the attention of the general research and technical communities.

1.4 ONE DATA CENTER VS. SEVERAL

In this book, we define the computer to be architected as one data center, even though many internet services use multiple data centers located far apart. Multiple data centers are sometimes used as complete replicas of the same service, with replication used mostly to reduce user latency and improve serving throughput. In those cases, a given user request is fully processed within one data center, and our machine definition seems appropriate.

In cases where a user query involves computation across multiple data centers, our single data center focus is a less obvious fit. Typical examples are services that deal with nonvolatile user data updates, and therefore require multiple copies for disaster tolerance. For such computations, a set of data centers might be the more appropriate system. Similarly, video streaming workloads benefit significantly from a content-distribution network (CDN) across multiple data centers and edge points.

However, we have chosen to think of the multi-data center scenario as more analogous to a network of computers. This is in part to limit the scope of this book, but also mainly because the huge gap in connectivity quality between intra- and inter-data center communications causes programmers to view such systems as separate computational resources. As the software development environment for this class of applications evolves, or if the connectivity gap narrows significantly in the future, we may need to adjust our choice of machine boundaries.

1.5 WHY WSCS MIGHT MATTER TO YOU

In the first edition of the book a decade ago, we discussed how WSCs might be considered a niche area, because their sheer size and cost render them unaffordable by all but a few large internet companies. We argued then that we did not believe this to be true, and that the problems that large internet services face would soon be meaningful to a much larger constituency because many organizations would be able to afford similarly sized computers at a much lower cost.

Since then our intuition has come true. Today, the attractive economics of low-end server class computing platforms puts clusters of thousands of nodes within the reach of a relatively broad range of corporations and research institutions. Combined with the trends around ever increasing numbers of processor cores on a single die, a single rack of servers today has more hardware threads than many data centers had a decade ago. For example, a rack with 40 servers, each with four 16-core dual-threaded CPUs, contains more than 4,000 hardware threads! Such systems exhibit the same scale, architectural organization, and fault behavior of WSCs from the last decade.[1]

[1] The relative statistics about sources of hardware faults can change substantially in these more integrated future systems, but silicon trends around less reliable components and the likely continuing high impact of software-driven faults suggest that programmers of such systems still need to deal with a fault-ridden platform.

However, perhaps more importantly, the explosive growth and popularity of Infrastructure-as-a-Service (IaaS) cloud computing offerings have now made WSCs available to anyone with a credit card [Arm+10]. We believe that our experience building these systems is useful in understanding the design issues and programming challenges for the next-generation cloud computing platform.

1.6 ARCHITECTURAL OVERVIEW OF WSCS

The hardware implementation of a WSC differs significantly from one installation to the next. Even within a single organization such as Google, systems deployed in different years use different basic elements, reflecting the hardware improvements provided by the industry. However, the architectural organization of these systems is relatively stable. Therefore, it is useful to describe this general architecture at a high level as it sets the background for subsequent discussions.

1.6.1 SERVERS

The hardware building blocks for WSCs are low-end servers, typically in a 1U[2] or blade enclosure format, and mounted within a rack and interconnected using a local Ethernet switch. These rack-level switches, which can use 40 Gbps or 100 Gbps links, have a number of uplink connections to one or more cluster-level (or data center-level) Ethernet switches. This second-level switching domain can potentially span more than 10,000 individual servers. In the case of a blade enclosure, there is an additional first level of networking aggregation within the enclosure where multiple processing blades connect to a small number of networking blades through an I/O bus such as PCIe. Figure 1.1(a) illustrates a Google server building block. More recently, WSCs have featured additional compute hardware building blocks, including GPUs and custom accelerators (for example, Figure 1.1(b) illustrates a TPU board). Similar to servers, these are connected through custom or industry-standard interconnects at the rack (or multi-rack *pod*) levels, leading up to the data center network. Figure 1.1(c) illustrates storage trays that are used to build out storage. Figure 1.2 shows how these building blocks are assembled into rows of racks of servers for both general-purpose servers and accelerators.

[2] Being satisfied with neither the metric nor the U.S. system, rack designers use “rack units” to measure the height of servers. 1U is 1.75 in or 44.45 mm; a typical rack is 42U high.

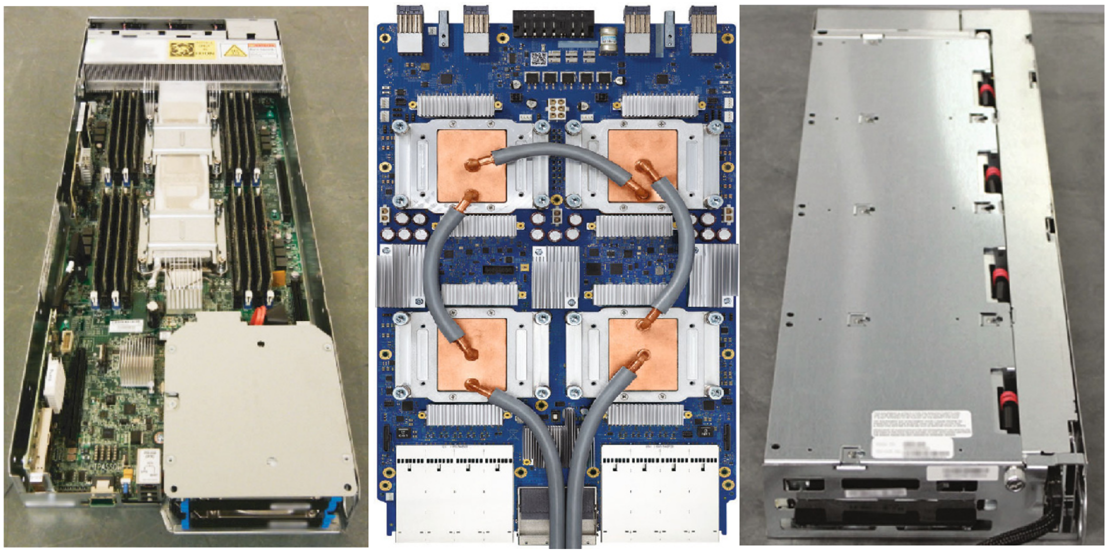
Figure 1.1: Example hardware building blocks for WSCs. Left to right: (a) a server board, (b) an accelerator board (Google's Tensor Processing Unit [TPU]), and (c) a disk tray.

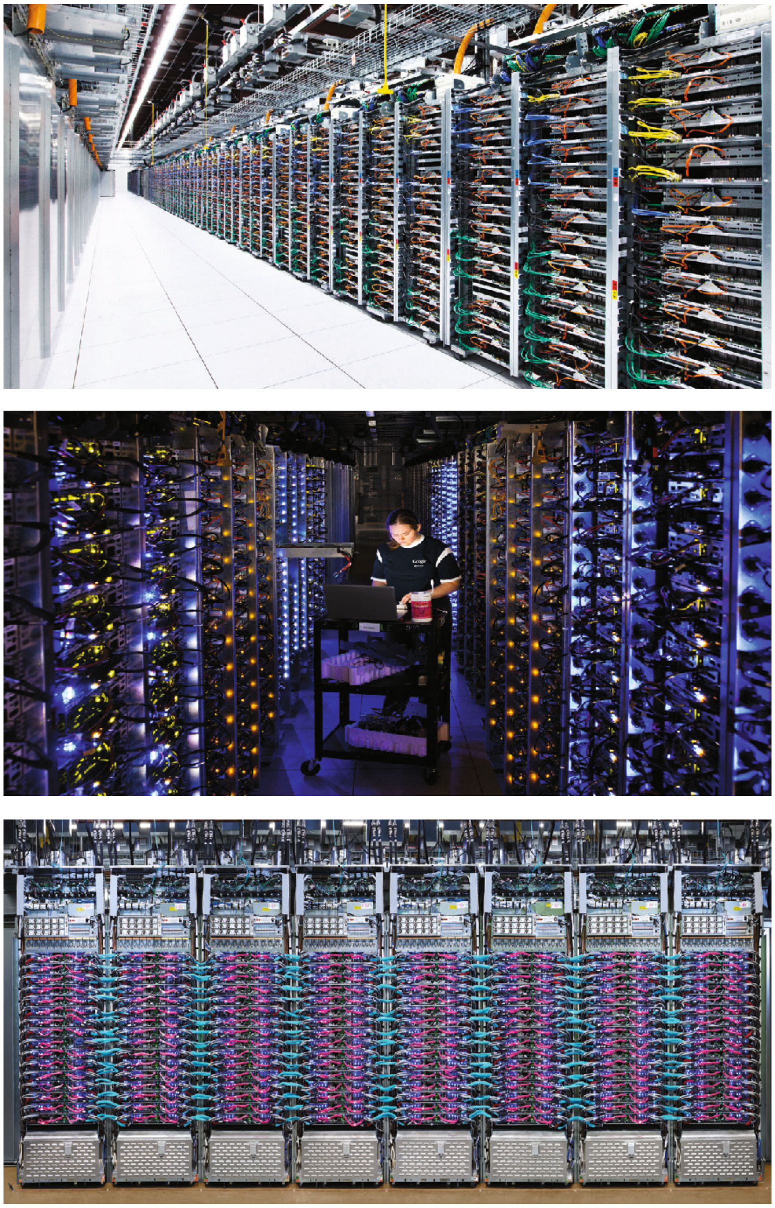

Figure 1.2: Hardware building blocks assembled into interconnected racks and rows.

1.6.2 STORAGE

Disks and Flash SSDs are the building blocks of today's WSC storage system. To provide durable storage to a large number of applications, these devices are connected to the data center network and managed by sophisticated distributed systems. WSC system designers need to make several tradeoffs based on their requirements. For example, should the disks and SSDs be connected directly to compute servers (Directly Attached Storage, or DAS) or disaggregated as part of Network Attached Storage (NAS)? While DAS can reduce hardware costs and improve network utilization (the network port is dynamically shared between compute and storage tasks), the NAS approach tends to simplify deployment and provide higher QoS to avoid performance interference from compute jobs. Alos, WSCs, including Google's, deploy desktop-class disk drives (or their close cousins, nearline drives) instead of enterprise-grade disks to save costs. Overall, storage devices in WSCs should aim at an aggregated view of the global optima across key metrics, including bandwidth, IOPS, capacity, tail latency, and TCO.

Distributed storage systems not only manage storage devices, but also provide unstructured and structured APIs for application developers. Google's Google File System (GFS), and later Colossus and its Cloud cousin GCS [Ser17], are examples of unstructured WSC storage systems that use space-efficient Reed-Solomon codes and fast reconstruction for high availability. Google's BigTable [Cha+06] and Amazon's Dynamo [DeC+07] are examples of structured WSC storage that provides database-like functionality but with weaker consistency models. To simplify developers' tasks, newer generations of structured storage systems such as Spanner [Cor+12] provide an SQL-like interface and strong consistency models.

The nature of distributed storage in WSCs also leads to the interplay of storage and networking technologies. The fast evolution and improvement of data center networking have created a large gap between network and disk performance, to the point that WSC designs can be dramatically simplified to not consider disk locality. On the other hand, low-latency devices such as Flash SSD and emerging Non-Volatile Memories (NVMs) pose new challenges for WSC design.

WSC designers need to build balanced systems with a hierarchy of memory and storage technologies, holistically considering the cluster-level aggregate capacity, bandwidth, and latency. Chapter 3 discusses system balance in more detail.

1.6.3 NETWORKING FABRIC

Choosing a networking fabric for WSCs involves a trade-off between speed, scale, and cost. As of 2018, it is not hard to find switches with 48 ports to interconnect servers at full speed of 40–100 Gbps Ethernet within a single rack. As a result, bandwidth within a rack of servers tends to be homogeneous. However, network switches with high port counts, which are needed to tie together WSC clusters, have a much different price structure and are more than 10 times more expensive (per port) than commodity rack switches. As a rule of thumb, a switch with 10 times the bisection bandwidth often costs about 100 times more. As a result of this cost discontinuity, the networking fabric of WSCs is often organized as a two-level hierarchy. Commodity switches in each rack provide a fraction of their bisection bandwidth for inter-rack communication through a handful of uplinks to the more costly cluster-level switches. For example, a 48-port rack-level switch could connect 40 servers to 8 uplinks, for a 5:1 oversubscription (8–20 Gbps per server uplink bandwidth). In this network, programmers must be aware of the relatively scarce cluster-level bandwidth resources and try to exploit rack-level networking locality, complicating software development and possibly impacting resource utilization.

Alternatively, one can remove some of the cluster-level networking bottlenecks by spending more money on the interconnect fabric. For example, Infiniband interconnects typically scale to a few thousand ports but can cost much more than commodity Ethernet on a per port basis. Similarly, some networking vendors are starting to provide larger-scale Ethernet fabrics, but again at a much higher cost per server. How much to spend on networking vs. additional servers or storage is an application-specific question that has no single correct answer. However, for now, we will assume that intra-rack connectivity is cheaper than inter-rack connectivity.

1.6.4 BUILDINGS AND INFRASTRUCTURE

So far, we have discussed the compute, storage, and network building blocks of a WSC. These are akin to the CPU, memory, disk, and NIC components in a PC. We still need additional components like power supplies, fans, motherboards, chassis, and other components, to make a full computer. Similarly, a WSC has other important components related to power delivery, cooling, and building infrastructure that also need to be considered.

WSC buildings (and campuses) house the computing, network, and storage infrastructure discussed earlier, and design decisions on the building design can dramatically influence the availability and uptime of the WSC. (Chapter 4 discusses the tier levels used in the data center construction industry.)

Similarly, WSCs have elaborate power delivery designs. At the scale that they operate, WSCs can often consume more power than thousands of individual households. Therefore, they use a holistic and hierarchical power delivery design that feeds electricity from the utility, to the substation, to power distribution units, to bus ducts, to individual power rails and voltage regulators on the server board, while also providing corresponding backup and redundancy such as uninterruptible power supplies (UPSs), generators, and backup batteries at different levels of the topology.

WSCs can also generate a lot of heat. Similar to power delivery, WSCs employ an elaborate end-to-end cooling solution with a hierarchy of heat-exchange loops, from circulated air cooled by fan coils on the data center floor, to heat exchangers and chiller units, all the way to cooling towers that interact with the external environment.

The building design, delivery of input energy, and subsequent removal of waste heat all drive a significant fraction of data center costs proportional to the amount of power delivered, and also have implications on the design and performance of the compute equipment (for example, liquid cooling for accelerators) as well as the availability service level objectives (SLOs) seen by workloads. These are therefore as important to optimize as the design of the individual compute, storage, and networking blocks.

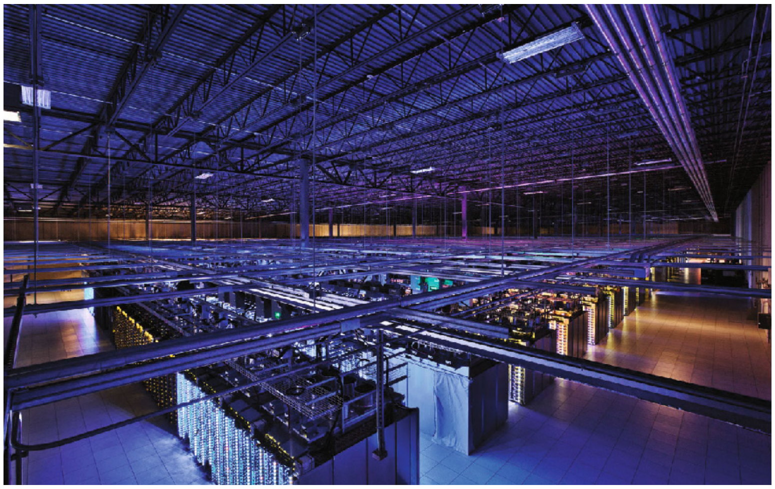

Figure 1.3: Power distribution, Council Bluffs, Iowa, U.S.

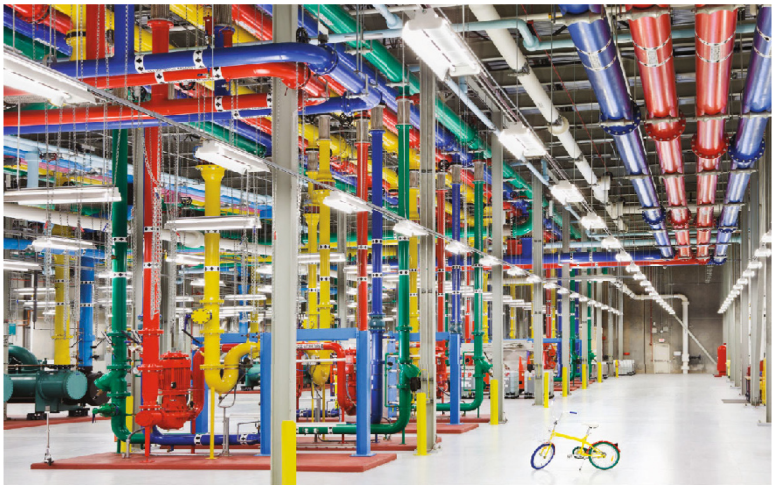

Figure 1.4: Data center cooling, Douglas County, Georgia, U.S.

Figure 1.5: Cooling towers and water storage tanks, Lenoir, North Carolina, U.S.

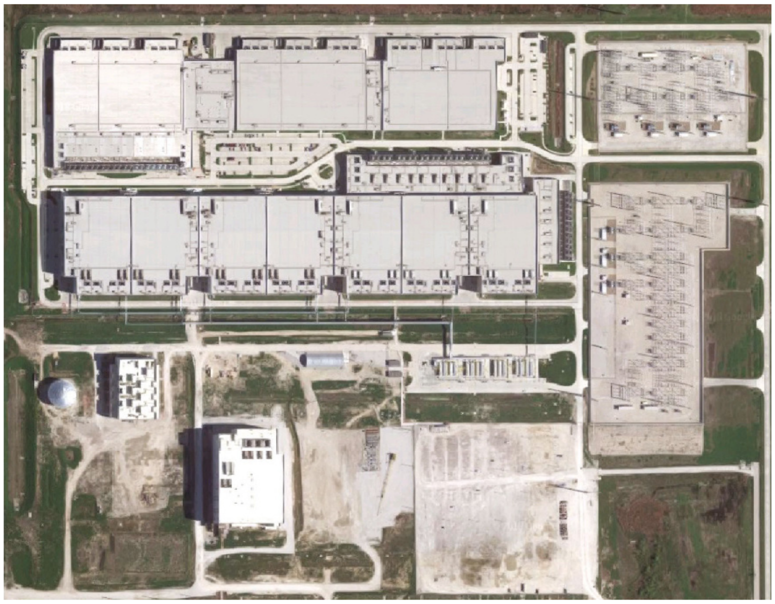

Figure 1.6: Aerial view of Google data center, Council Bluffs, Iowa, U.S.

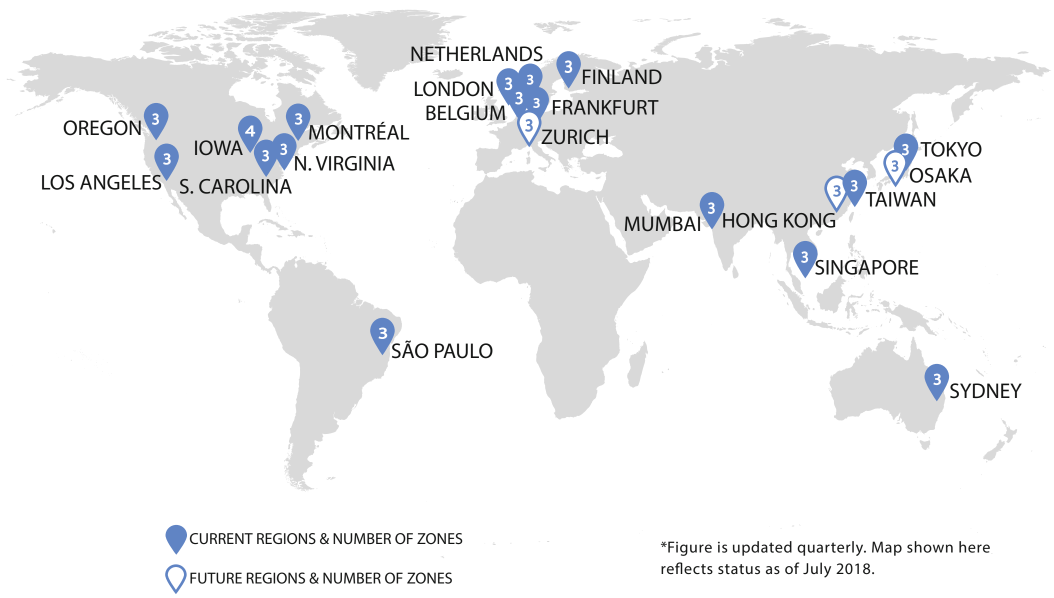

Figure 1.7: Google Cloud Platform regions and number of zones, circa July 2018. The latest source is available at https://cloud.google.com/about/locations/.

1.6.5 POWER USAGE

Energy and power usage are important concerns in the design of WSCs because, as we discuss in more detail in Chapter 5, energy-related costs have become an important component of the total cost of ownership of this class of systems. Figure 1.8 provides some insight into how energy is used in modern IT equipment by breaking down the peak power usage of one generation of WSCs deployed at Google in 2017 by main component group.

Although this breakdown will vary significantly depending on how systems are configured for a given workload, the graph indicates that CPUs are the dominant energy consumer in WSCs. Interestingly, the first edition of this book showed the relative energy use of the memory system rising to near parity with CPU consumption. Since then that trend has reversed due to a combination of effects. First, sophisticated thermal management has allowed CPUs to run closer to their maximum power envelope, resulting in higher energy consumption per CPU socket. Second, memory technology has shifted away from power hungry FBDIMMs to DDR3 and DDR4 systems with better energy management. Third, DRAM voltage has dropped from 1.8 V down to 1.2 V. Finally, today's systems have a higher ratio of CPU performance per gigabyte of DRAM, possibly as a result of a more challenging technology scaling roadmap for main memory. While increasing bandwidth demands can still reverse these trends, currently, memory power is still significantly smaller than CPU power. It is also worth noting that the power and cooling overhead are relatively small, reflecting generations of improvements in this area, many specific to WSCs. In Chapter 5, we discuss WSC energy efficiency in further detail; see Figure 5.6 for a discussion on power usage *as a function of load*.

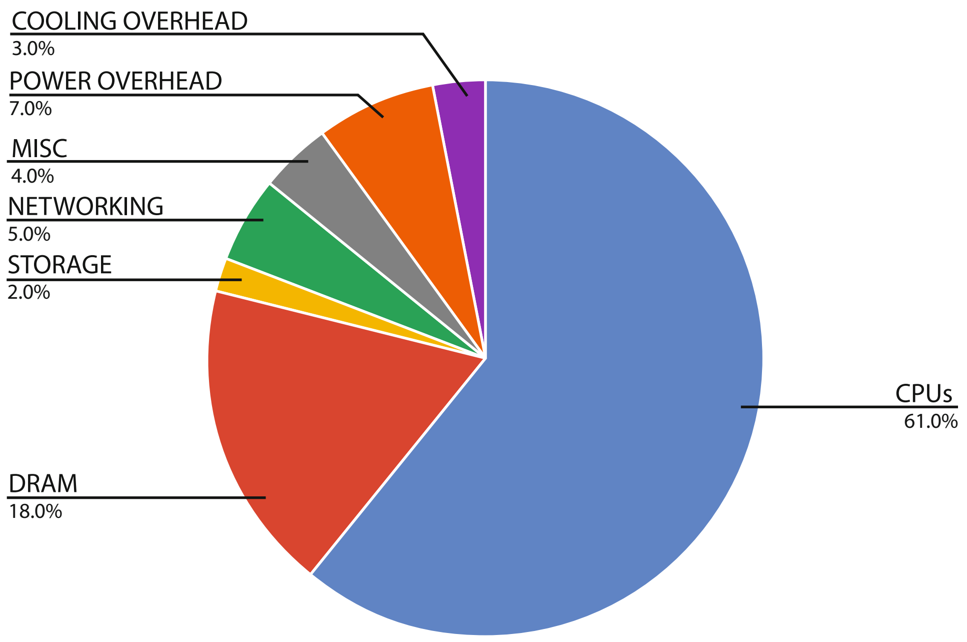

Figure 1.8: Approximate distribution of peak power usage by hardware subsystem in a modern data center using late 2017 generation servers. The figure assumes two-socket x86 servers and 12 DIMMs per server, and an average utilization of 80%.

1.6.6 HANDLING FAILURES AND REPAIRS

The sheer scale of WSCs requires that internet services software tolerates relatively high component fault rates. Disk drives, for example, can exhibit annualized failure rates higher than 4% [PWB07; SG07b]. Different deployments have reported between 1.2 and 16 average server-level restarts per year. With such high component failure rates, an application running across thousands of machines may need to react to failure conditions on an hourly basis. We expand on this topic in Chapter 2, which describes the application domain, and Chapter 7, which deals with fault statistics.

1.7 OVERVIEW OF BOOK

We will elaborate on the issues discussed above in the rest of the book.

Chapter 2 starts with an overview of applications that run on WSCs and that define all the later system design decisions and trade-offs. We discuss key applications like web search and video streaming, and also cover the systems infrastructure stack, including platform-level software, cluster-level infrastructure, and monitoring and management software.

Chapter 3 covers the key hardware building blocks. We discuss the high-level design considerations in WSC hardware and focus on server and accelerator building blocks, storage architectures, and data center networking designs. We also discuss the interplay between compute, storage, and networking, and the importance of system balance.

Chapter 4 looks at the next level of system design, focusing on data center power, cooling infrastructure, and building design. We provide an overview of the basics of the mechanical and electrical engineering involved in the design of WSCs and delve into case studies of how Google designed the power delivery and cooling in some of its data centers.

Chapter 5 discusses the broad topics of energy and power efficiency. We discuss the challenges with measuring energy efficiency consistently and the power usage effectiveness (PUE) metric for data center-level energy efficiency, and the design and benefits from power oversubscription. We discuss the energy efficiency challenges for computing, with specific focus on energy proportional computing and energy efficiency through specialization.

Chapter 6 discusses how to model the total cost of ownership of WSC data centers to address both capital expenditure and operational costs, with case studies of traditional and WSC computers and the trade-offs with utilization and specialization.

Chapter 7 discusses uptime and availability, including data that shows how faults can be categorized and approaches to dealing with failures and optimizing repairs.

Chapter 8 concludes with a discussion of historical trends and a look forward. With the slowing of Moore's Law, we are entering an exciting era of system design, one where WSC data centers and cloud computing will be front and center, and we discuss the various challenges and opportunities ahead.