The Computing Landscape of the 21st Century

Mahadev Satyanarayanan
Carnegie Mellon University
satya@cs.cmu.edu

Wei Gao
University of Pittsburgh
WEIGAO@pitt.edu

Brandon Lucia
Carnegie Mellon University
blucia@andrew.cmu.edu

ABSTRACT

This paper shows how today's complex computing landscape can be understood in simple terms through a 4-tier model. Each tier represents a distinct and stable set of design constraints that dominate attention at that tier. There are typically many alternative implementations of hardware and software at each tier, but all of them are subject to the same set of design constraints. We discuss how this simple and compact framework has explanatory power and predictive value in reasoning about system design.

ACM Reference Format:
Mahadev Satyanarayanan, Wei Gao, and Brandon Lucia. 2019. The Computing Landscape of the 21st Century. In *The 20th International Workshop on Mobile Computing Systems and Applications (HotMobile '19), February 27–28, 2019, Santa Cruz, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3301293.3302357

1 Introduction

The creation of the Periodic Table in the late nineteenth and early twentieth centuries was an exquisite intellectual feat [36]. In a small and simple data structure, it organizes our knowledge about all the elements in our universe. The position of an element in the table immediately suggests its physical attributes and its chemical affinities to other elements. The presence of "holes" in early versions of the table led to the search and discovery of previously unknown elements with predicted properties. This simple data structure has withstood the test of time. As new man-made elements were created, they could all be accommodated within the existing framework. The quest to understand the basis of order in this table led to major discoveries in physics and chemistry. The history of the periodic table teaches us that there is high value in distilling and codifying taxonomical knowledge into a compact form.

Today, we face a computing landscape of high complexity that is reminiscent of the scientific landscape of the late 19th century. Is there a way to organize our computing universe into a simple and compact framework that has explanatory power and predictive value? What is our analog of the periodic table? In this paper, we describe our initial effort at such an intellectual distillation. The periodic table took multiple decades and the contributions of many researchers to evolve into the familiar form that we know today. We therefore recognize that this paper is only the beginning of an important conversation in the research community.

HotMobile '19, February 27–28, 2019, Santa Cruz, CA, USA

ACM ISBN 978-1-4503-6273-3/19/02.
https://doi.org/10.1145/3301293.3302357

2 A Tiered Model of Computing

Today's computing landscape is best understood by the *tiered model* shown in Figure 1. Each tier represents a distinct and stable set of design constraints that dominate attention at that tier. There are typically many alternative implementations of hardware and software at each tier, but all of them are subject to the same set of design constraints. There is no expectation of full interoperability across tiers — randomly choosing one component from each tier is unlikely to result in a functional system. Rather, there are many sets of compatible choices across tiers. For example, a single company will ensure that its products at each tier work well with its own products in other tiers, but not necessarily with products of other companies. The tiered model of Figure 1 is thus quite different from the well-known "hourglass" model of interoperability. Rather than defining functional boundaries or APIs, the tiered model segments the end-to-end computing path and highlights design commonalities.

In each tier there is considerable churn at timescales of up to a few years, driven by technical progress as well as market-driven tactics and monetization efforts. The relationship between tiers, however, is stable over decade-long timescales. A major shift in computing typically involves the appearance, disappearance or re-purposing of a tier in Figure 1. We describe the four tiers of Figure 1 in the rest of this section. Section 3 then explains how the tiered model can be used as an aid to reasoning about the design of a distributed system. Section 4 examines energy relationships across tiers. Section 5 interprets the past six decades of computing in the context of Figure 1, and Section 6 speculates on the future.

2.1 Tier-1: Elasticity, Permanence and Consolidation

Tier-1 represents "the cloud" in today's parlance. Two dominant themes of Tier-1 are *compute elasticity* and *storage permanence.* Cloud computing has almost unlimited elasticity, as a Tier-1 data center can easily spin up servers to rapidly meet peak demand. Relative to Tier-1, all other tiers have very limited elasticity. In terms of archival preservation, the cloud is the safest place to store data with confidence that it can be retrieved far into the future. A combination of storage redundancy (e.g., RAID), infrastructure stability (i.e., data center engineering), and management practices (e.g., data backup and disaster recovery) together ensure the long-term integrity and accessibility of data entrusted to the cloud. Relative to the data permanence of Tier-1, all other tiers offer more tenuous safety. Getting important data captured at those tiers to the cloud is often an imperative. Tier-1 exploits economies of scale to offer very low total costs of computing. As hardware costs shrink relative to personnel costs, it becomes valuable to amortize IT personnel costs over many machines in a large data center. *Consolidation* is thus a third dominant theme of Tier-1. For large tasks without strict timing, data ingress volume, or data privacy requirements, Tier-1 is typically the optimal place to perform the task.

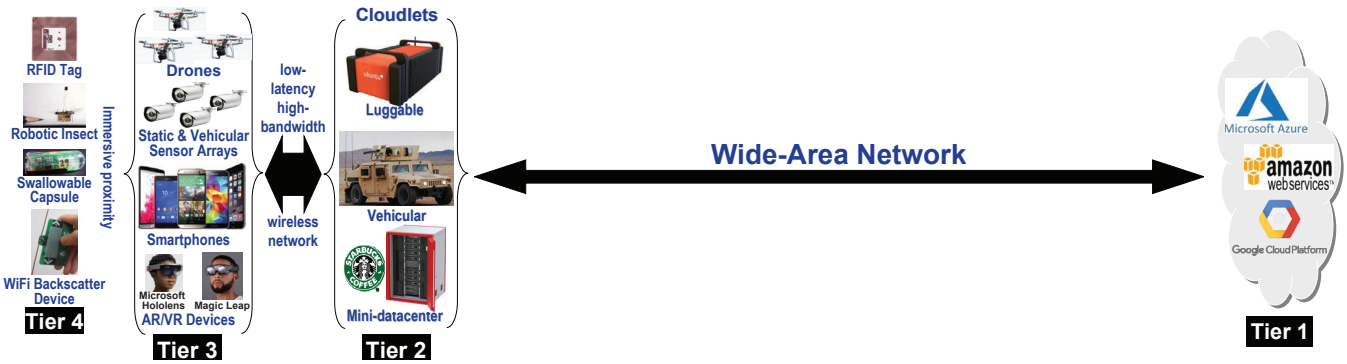


Figure 1: Four-tier Model of Computing

2.2 Tier-3: Mobility and Sensing

We consider Tier-3 next, because understanding its attributes helps to define Tier-2. *Mobility* is a defining attribute of Tier-3 because it places stringent constraints on weight, size, and heat dissipation of devices that a user carries or wears [29]. Such a device cannot be too large, too heavy or run too hot. Battery life is another crucial design constraint. Together, these four constraints severely limit designs at Tier-3. Technological breakthroughs (e.g., a new battery technology or a new lightweight and flexible display material) may expand the envelope of designs, but the underlying constraints always remain.

Sensing is another defining attribute of Tier-3. Today's mobile devices are rich in sensors such as GPS, microphones, accelerometers, gyroscopes, and video cameras. Unfortunately, a mobile device may not be powerful enough to perform real-time analysis of data captured by its on-board sensors (e.g., video analytics). While mobile hardware continues to improve, there is always a large gap between what is feasible on a mobile device and what is feasible on a server of the same technological era. Figure 2 shows this large performance gap persisting over a 20-year period from 1997 to 2017. One can view this stubborn gap as a "mobility penalty" — i.e., the price one pays in performance foregone in order to meet mobility constraints.

To overcome this penalty, a mobile device can offload computation over a wireless network to Tier-1. This was first described by Noble et al [25] in 1997, and has since been extensively explored by many others [8, 32]. For example, speech recognition and natural language processing in iOS and Android nowadays work by offloading their compute-intensive aspects to the cloud.

IoT devices can be viewed as Tier-3 devices. Although they may not be mobile, there is a strong incentive for them to be inexpensive. Since this typically implies meager processing capability, offloading computation to Tier-1 is again attractive.

2.3 Tier-2: Network Proximity

As mentioned in Section 2.1, economies of scale are achieved in Tier-1 by consolidation into a few very large data centers. Extreme consolidation has two negative consequences. First, it tends to lengthen network round-trip times (RTT) to Tier-1 from Tier-3 — if there are very few Tier-1 data centers, the closest one is likely to be far away. Second, the high fan-in from Tier-3 devices implies high cumulative ingress bandwidth demand into Tier-1 data centers. These negative consequences stifle the emergence of new classes of real-time, sensor-rich, compute-intensive applications [34].

	Typical Tier-1 Server		Typical Tier-3 Device	
Year	Processor	Speed	Device	Speed
1997	Pentium II	266 MHz	Palm Pilot	16 MHz
2002	Itanium	1 GHz	Blackberry 5810	133 MHz
2007	Intel Core 2	9.6 GHz (4 cores)	Apple iPhone	412 MHz
2011	Intel Xeon X5	32 GHz (2x6 cores)	Samsung Galaxy S2	2.4 GHz (2 cores)
2013	Intel Xeon E5-2697v2	64 GHz (2x12 cores)	Samsung Galaxy S4	6.4 GHz (4 cores)
			Google Glass	2.4 GHz (2 cores)
2016	Intel Xeon E5-2698v4	88.0 GHz (2x20 cores)	Samsung Galaxy S7	7.5 GHz (4 cores)
			HoloLens	4.16 GHz (4 cores)
2017	Intel Xeon Gold 6148	96.0 GHz (2x20 cores)	Pixel 2	9.4 GHz (4 cores)

Source: Adapted from Chen [3] and Flinn [8]
"Speed" metric = number of cores times per-core clock speed.

Figure 2: The Mobility Penalty: Impact of Tier-3 Constraints

Tier-2 addresses these negative consequences by creating the illusion of bringing Tier-1 "closer." This achieves two things. First, it enables Tier-3 devices to offload compute-intensive operations at very low latency. This helps to preserve the tight response time bounds needed for immersive user experience (e.g., augmented reality (AR)) and cyber-physical systems (e.g., drone control). Proximity also results in a much smaller fan-in between Tiers-3 and -2 than is the case when Tier-3 devices connect directly to Tier-1. Consequently, Tier-2 processing of data captured at Tier-3 avoids excessive bandwidth demand anywhere in the system. Server hardware at Tier-2 is essentially the same as at Tier-1 (i.e., the second column of Figure 2), but engineered differently. Instead of extreme consolidation, servers in Tier-2 are organized into small, dispersed data centers called *cloudlets*. A cloudlet can be viewed as "a data center in a box." When a Tier-3 component such as a drone moves far from its current cloudlet, a mechanism analogous to cellular handoff is required to discover and associate with a new optimal cloudlet [9]. The introduction of Tier-2 is the essence of *edge computing* [33].

Note that "proximity" here refers to *network proximity* rather than physical proximity. It is crucial that RTT be low and end-to-end bandwidth be high. This is achievable by using a fiber link between a wireless access point and a cloudlet that is many tens or even hundreds of kilometers away. Conversely, physical proximity does not guarantee network proximity. A highly congested WiFi network may have poor RTT, even if Tier-2 is physically near Tier-3.

2.4 Tier-4: Longevity and Opportunism

A key driver of Tier-3 is the vision of *embedded sensing,* in which tiny sensing-computing-communication platforms continuously report on their environment. "Smart dust" is the extreme limit of this vision. The challenge of cheaply maintaining Tier-3 devices in the field has proved elusive because replacing their batteries or charging them is time-consuming and/or difficult.

This has led to the emergence of devices that contain no chemical energy source (battery). Instead, they harvest incident EM energy (e.g., visible light or RF) to charge a capacitor, which then powers a brief episode of sensing, computation and wireless transmission. The device then remains passive until the next occasion when sufficient energy can be harvested to power another such episode. This modality of operation, referred to as *intermittent computing* [17, 18, 21], eliminates the need for energy-related maintenance of devices in the field. This class of devices constitutes Tier-4 in the taxonomy of Figure 1. *Longevity* of deployment combined with *opportunism* in energy harvesting are the distinctive attributes of this tier.

The most successful Tier-4 devices today are RFID tags, which are projected to be a roughly $25 billion market by 2020 [27]. More sophisticated devices are being explored in research projects including, for example, a robotic flying insect powered solely by an incident laser beam [12]. A Tier-3 device (e.g., RFID reader) provides the energy that is harvested by a Tier-4 device. *Immersive proximity* is thus the defining relationship between Tier-4 and Tier-3 devices — they have to be physically close enough for the Tier-4 device to harvest sufficient energy for an episode of intermittent computation. Network proximity alone is not sufficient. RFID readers have a typical range of a few meters today. A Tier-4 device stops functioning when its energy source is misaimed or too far away.

3 Using the Model

The tiers of Figure 1 can be viewed as a canonical representation of components in a modern distributed system. Of course, not every distributed system will have all four tiers. For example, a team of users playing Pokemon Go will only use smartphones (Tier-3) and a server in the cloud (Tier-1). A worker in a warehouse who is taking inventory will use an RFID reader (Tier-3) and passive RFID tags that are embedded in the objects being inventoried (Tier-4). A more sophisticated design of this inventory control system may allow multiple users to work concurrently, and to use a cloudlet in the warehouse (Tier-2) or the cloud (Tier-1) to do aggregation and duplicate elimination of objects discovered by different workers. In general, one can deconstruct any complex distributed system and then examine the system from a tier viewpoint. Such an analysis can be a valuable aid to deeper understanding of the system.

As discussed in Section 2, each tier embodies a small set of salient properties that define the reason for the existence of that tier. Elasticity, permanence and consolidation are the salient attributes of Tier-1; mobility and sensing are those of Tier-3; network proximity to Tier-3 is the central purpose of Tier-2; and, longevity combined with opportunism represents the essence of Tier-4. These salient attributes shape both hardware and software designs that are relevant to each tier. For example, hardware at Tier-3 is expected to be mobile and sensor-rich. Specific instances (e.g., a static array of video cameras) may not embody some of these attributes (i.e., mobility), but the broader point is invariably true. The salient attributes of a

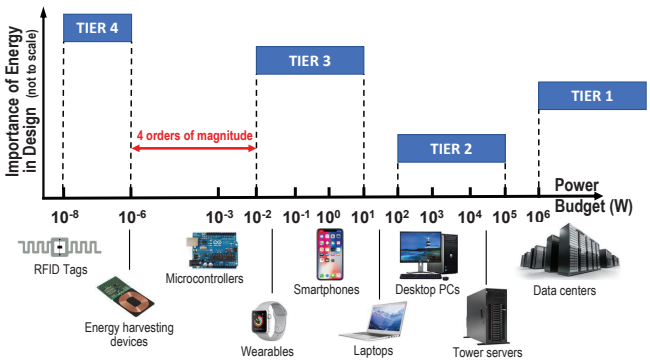


Figure 3: Importance of Energy as a Design Constraint

tier severely constrain its range of acceptable designs. A mobile device, for example, has to be small, lightweight, energy-efficient and have a small thermal footprint. This imperative follows directly from the salient attribute of mobility. A product that does not meet this imperative will simply fail in the marketplace.

The same reasoning also applies to software at each tier. For example, Tier-3 to Tier-1 communication is (by definition) over a WAN and may involve a wireless first hop that is unreliable and/or congested. Successful Tier-3 software design for this context has to embody support for disconnected and weakly-connected operation. On the other hand, Tier-3 to Tier-2 communication is expected to be LAN or WLAN quality at all times. A system composed of just those tiers can afford to ignore support for network failure. Note that the server hardware in the two cases may be identical: located in a data center (Tier-1) or in a closet nearby (Tier-2). It is only the placement and communication assumptions that are different.

Constraints serve as valuable discipline in system design. Although implementation details of competing systems with comparable functionality may vary widely, their tier structure offers a common viewpoint from which to understand their differences. Some design choices are forced by the tier, while other design choices are made for business reasons, for compatibility reasons with products in other tiers, for efficiency, usability, aesthetics, and so on. Comparison of designs from a tier viewpoint helps to clarify and highlight the essential similarities versus incidental differences.

Like the periodic table mentioned in Section 1, Figure 1 distills a vast space of possibilities (i.e., design choices for distributed systems) into a compact intellectual framework. However, the analogy should not be over-drawn since the basis of order in the two worlds is very different. The periodic table exposes order in a "closed-source" system (i.e., nature). The tiered model reveals structure in an "open-source" world (i.e., man-made system components). The key insight of the tiered model is that, in spite of all the degrees of freedom available to designers, the actual designs that thrive in the real world have deep structural similarities.

4 The Central Role of Energy

A hidden message of Section 2 is that *energy* plays a central role in segmentation across tiers. As shown in Figure 3, the power concerns at different tiers span many orders of magnitude, from a few nanowatts (e.g., a passive RFID tag) to tens of megawatts (e.g., an exascale data center). Energy is also the most critical factor when making design choices in other aspects of a computing system. For

example, the limited availability of energy could severely limit performance. The power budget of a system design could also be a major barrier to reductions of system cost and form factor. The relative heights of tiers in Figure 3 are meant to loosely convey the extent of energy's influence on design at that tier.

Tier-1 (Data Centers): Power is used in a data center for IT equipment (e.g., servers, networks, storage, etc) and infrastructure (e.g., cooling systems), adding up to as much as 30 MW at peak hours [6]. Current power saving techniques focus on load balancing and dynamically eliminating power peaks. Power oversubscription enables more servers to be hosted than theoretically possible, leveraging the fact that their peak demands rarely occur simultaneously.

Tier-2 (Cloudlets): Cloudlets can span a wide range of form factors, from high-end laptops and desktop PCs to tower or rack servers. Power consumption can therefore vary from <100W to several kilowatts. At this tier, well-known power saving techniques such as CPU frequency scaling [14] are applicable. Techniques have also been developed to reduce the power consumption of attached hardware (e.g., GPUs) [23], and to balance the power consumption among multiple interconnected cloudlets. As Figure 3 suggests, energy constraints are relatively easy to meet at this tier.

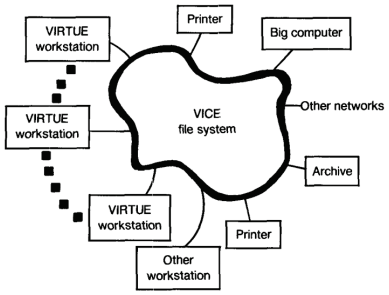
Tier-3 (Smartphones): Smartphones are the dominant type of computing device at Tier-3. Their power consumption is below 1000 mW when idle [22], but can peak at 3500-4000 mW. Techniques such as frequency scaling, display optimization, and application partitioning are used to reduce power consumption. Workload-specific techniques are also used in web browsing and mobile gaming.

Tier-3 (Wearables): Studies have shown that the energy consumption of smartwatches can be usually controlled to below 100 mW in stand-by mode with screen off [16]. When the screen is on or the device is wirelessly transmitting data, the energy consumption could surge to 150-200 mW. Various techniques have been proposed to further reduce smartwatch power consumption to <100 mW in active modes via energy-efficient storage or display management.

Tier-4: Energy-harvesting enables infrastructure-free, low-maintenance operation for tiny devices that sense, compute and communicate. Energy harvesting presents unique challenges: sporadic power is limited to 10^{-7} to 10^{-8} watts using, e.g., RF or biological sources. A passive RFID tag consumes hundreds of nA at 1.5V [26]. Emerging wireless backscatter networking enables communication at extremely low power [15]. Intermittent computing allows sensing and complex processing on scarce energy [5, 17]. Such capabilities enable a new breed of sensors and actuators deployed in the human body to monitor health signals, in civil infrastructure, and in adversarial environments like outer space [5]. RF beamforming extends the capability of batteryless, networked, in-vivo devices [19].

5 A Tiered View of the Past

In the beginning, there was only Tier-1. The batch-processing mainframes of the late 1950s and 1960s represented consolidation in its extreme form. In this primitive world, there were no representatives of Tier-2, Tier-3 or Tier-4. Those tiers could not emerge until the hardware cost of computing and its physical size had dropped by many orders of magnitude. The wide-area network shown in Figure 1 did not exist, but it was foreshadowed by remote punch card readers and line printers connected via point-to-point links to a mainframe.



User mobility is supported: A user can walk to any workstation in the system and access any file in the shared name space. A user's workstation is personal only in the sense that he owns it.

System administration is easier: Operations staff can focus on the relatively small number of servers, ignoring the more numerous and physically dispersed clients. Adding a new workstation involves merely connecting it to the network and assigning it an address.

The figure at the top is from the 1986 description of the Andrew project by Morris et al [24]. The cloud-like Tier-1 entity ("VICE file system") offers storage permanence for the Tier-2 entities at the periphery ("VIRTUE workstations"). The verbatim comments about mobility and system administration are from a 1990 paper [28] about this model of computing.

Figure 4: Limited Re-Creation of Tier-1 in a Tier-2 World

The emergence of timesharing by the late 1960s introduced elasticity to Tier-1. In a batch-processing system, a job was queued, and eventually received exclusive use of the mainframe. Queueing delays increased as more jobs competed for the mainframe, thereby exposing the inelasticity of this computing resource. Timesharing multiplexed the mainframe at fine granularity, rather than serially reusing it. It leveraged human think times to provide the illusion that each user had exclusive access to Tier-1. This illusion broke down at very high load by the increase in queueing delays for user interactions. Until that breaking point, however, Tier-1 appeared elastic to varying numbers of users. The introduction of virtual machine (VM) technology by the late 1960s expanded this illusion beyond user-level code. Now, elasticity applied to the entire vertical stack from low-level device drivers, through the (guest) operating system, to the top of the application stack. Many decades later, this encapsulating ability led to the resurgence of VMs in cloud computing.

Frustration with the queueing delays of timesharing led to the emergence of personal computing. In this major shift, Tier-1 was completely replaced by the brand-new Tier-2. An enterprise that switched from timesharing to personal computing was effectively disaggregating its consolidated Tier-1 infrastructure into a large number of dispersed Tier-2 devices. The dedication of a Tier-2 device to each user, combined with its physical proximity to the user, led to crisp interactive response. This, in turn, led to the emergence of a new class of latency-sensitive applications such as spreadsheets. A spreadsheet does not seem latency-sensitive today, but in the early 1980s its latency constraints could only be met at Tier-2.

An unintended consequence of the disaggregation of Tier-1 into dispersed Tier-2 elements was its negative impact on shared data. By the early 1980s, the archival data stored in its computing system was often of high value to an enterprise. Over the previous decade, business practices had been transformed by the easy sharing of data across timesharing users in an enterprise. The disaggregation of Tier-1 into dispersed Tier-2 devices destroyed the mechanisms for data sharing across users (e.g., a shared file system). It was at this

juncture that the third important attribute of Tier-1, namely storage permanence, came to be recognized as crucial. How to preserve Tier-1's ability to share information easily, securely, and with appropriate access controls in a dispersed and fragmented Tier-2 world became a major challenge. The Andrew project [24] addressed this challenge by re-creating Tier-1 for the limited purpose of storage permanence, as shown in Figure 4. A distributed file system (AFS [10, 35]) created the illusion that all of Tier-1 storage was accessible via on-demand caching at Tier-2 devices. The resulting system provided users with the ease of data sharing characteristic of Tier-1, while preserving the crisp interactive response of Tier-2. Today, systems such as DropBox and Box are modern realizations of this concept.

As discussed in Section 2.1, a key attribute of a modern Tier-1 data center is its large pool of compute nodes that provide elasticity. This capability was pioneered by the Cambridge Processor Bank [1] in the 1979-1988 time period, and a few years later by Amoeba [40].

The emergence of Tier-3 coincided with the release of the earliest computers (circa 1983) that were small enough to be considered portable devices. The Radio Shack TRS-80 Model 100 (weighing roughly 1.5 kg and powered by 4 AA batteries) and the Compaq Portable (weighing 13 kg) were two early examples. There was explosive innovation in laptop hardware by the late 1980s. Once the Internet became widely used (mid-1990s), a key distinction between Tier-2 and Tier-3 was the stability and quality of Internet connectivity. In contrast to Tier-2 devices, Tier-3 devices typically had wireless connectivity with periods of disconnection and poor connectivity. The desire to preserve shared enterprise data access even when mobile led to the creation of Coda File System [31], which extended Figure 4 to Tier-3 devices.

By the mid-1990s handheld mobile devices referred to as *personal digital assistants (PDAs)* emerged. In the same timeframe, computing hardware had become small and light enough for *wearable computers* to be created [38]. These extreme optimizations of Tier-3 devices led to the mobility penalty discussed earlier (Section 2.2 and Figure 2). The need to process sensor streams in real time from these devices led to offloading (originally called "cyber foraging" [30]) from Tier-3 to Tier-1 or Tier-2.

6 Future Evolution

Our future computing landscape will include computing modalities that are not covered by Figure 1. We speculate on these modalities and their implications in this section.

Biological Computer Systems: Future computer systems will be inspired by, rendered in, and extensive of biology. Neuromorphic computing is seeing a resurgance with analog [11, 37] and delay-based [20, 39] architectures for neural machine learning. While analogies to biological behavior abound, spanning from circuits, to architectures, to software and algorithms, computer system behavior is rarely biological in its efficiency and capability. Other emerging systems leverage the efficiency of biological systems, rendering molecular-scale data storage [2] and processing [13] structures directly in biological substrates, such as engineered DNA. A key advantage of engineered biological computing systems is their extremely high degree of parallelism, distributing the responsibility for a task across vast numbers of molecular-scale components. A key challenge is the lack of reliability of individual components. This can be mitigated by using the extreme parallelism for redundancy. Future computing systems will extend biology with the mechanical capabilities of micro- and nano-robotics. This extension can lead to an inversion of the relative costs of computing and actuation [7]. At macro-scale, the energy cost of actuation dominates that of computing; but at nano-scale, computing may dominate. Optimizing computing in such systems may lead to a new Tier-5.

Blurring boundaries between tiers: Tier boundaries in Figure 1 are likely to blur, leading to a continuum of devices with different power budgets, computing workloads and manufacturing costs. The major drivers of such blurring are advances in the manufacturing technologies. Such improvement not only allows a device to undertake more computing workloads with a lower power budget and a smaller form factor, but also fosters new computing models that fully integrate heterogeneous computing devices into a universal ecosystem. For example, significant chip-level convergence has occured across desktop PCs, laptops and smartphones in the past few years, leading to simpler task migration across these devices.

A consequence of such blurring boundaries is that the gap in capabilities between cloudlets at Tier 2 and battery-powered mobile devices at Tier 3 will diminish. In addition, today's requirement of chemical batteries at Tier-3 is likely to be gradually relaxed due to more advanced energy harvesting and wireless charging technologies. This allows significant reduction of device size and alleviates some design constraints of mobility. Energy harvesting will also be able to provide a much higher power budget, which then allows a richer set of computing tasks being executed at Tier-4 devices. Consequently, the transition between Tier 3 and Tier 4 will be much smoother. This smooth transition will lead to more convenient deployment of embedded computing objects and enable many new computing paradigms, such as distributed AI in the future IoT.

Quantum Computing: In terms of physical size, energy demand and dependence on external cooling, quantum computers would seem to map naturally to Tier-1. However, their widespread adoption requires extensive rework of the entire computing landscape. This includes new programming languages to express probabilistic problems, new programmable quantum architectures, new compilers that target emerging quantum machines, and new debugging and verification tools that validate results [4]. We do not yet know how to incorporate such a disruptive change into the tiered model of Figure 1. For now, quantum computing is an outlier.

7 Closing Thoughts

We began with the question "Is there a way to organize our computing universe into a simple and compact framework that has explanatory power and predictive value?" In Sections 2 through 6 we have presented and discussed such a framework. Its essence is the segmentation of the computing landscape into tiers that embody a set of design constraints and architectural roles. Figure 1 illustrates the four tiers that constitute our computing landscape today. The discussion of the past in Section 5 shows that these tiers are not pre-ordained. Rather, starting with a single tier, they have emerged over time in response to technical innovations and expanding goals.

Space, time and energy are the driving forces of this evolution. Networking, in general, and the Internet, in particular, grew out of our desire to transcend space. Mobility and proximity, which are both space-derived concepts, directly influence the designs of Tier-2, Tier-3 and Tier-4. Permanent storage aims to preserve precious data in

spite of the ravages of time. As shown in Figure 4, it accounts for the enduring role of Tier-1. Energy plays a central role in shaping tiers, as shown in Figure 3 and discussed in Section 4. These fundamental themes of space, time and energy will continue to shape computing long after today's technology is obsolete.

We close by reiterating that this paper only represents a first step, not the last word, in the creation of a compact intellectual framework for reasoning about design choices in distributed systems. The periodic table, for example, is able to resolve its structure into orthogonal axes of periods and groups. If a comparable resolution of Figure 1 into fundamental axes were possible, that would enhance the analytical and predictive power of the model. A different extension would be to incorporate a taxonomy of communication that amplifies the computing-centric taxonomy that we have introduced here. Much work remains to be done.

Acknowledgements

We thank our shepherd, Aruna Balasubramanian, and the anonymous reviewers for helping us to improve this work. This research was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0051 and by the National Science Foundation (NSF) under grant number CNS-1518865. Additional support was provided by Intel, Vodafone, Deutsche Telekom, Verizon, Crown Castle, NTT, and the Conklin Kistler family fund. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view(s) of their employers or the above-mentioned funding sources.

REFERENCES

[1] Jean M. Bacon, Ian M. Leslie, and Roger M. Needham. 1989. *Distributed computing with a processor bank*. Technical Report UCAM-CL-TR-168. University of Cambridge.

[2] James Bornholt, Randolph Lopez, Douglas M. Carmean, Luis Ceze, Georg Seelig, and Karin Strauss. 2016. A DNA-Based Archival Storage System. In *Proc. of the 21st Intl. Conf. on Architectural Support for Programming Languages and Operating Systems*. Atlanta, GA.

[3] Zhuo Chen. 2018. *An Application Platform for Wearable Cognitive Assistance*. Ph.D. Dissertation. Computer Science Dept., Carnegie Mellon Univ.

[4] Frederic T. Chong, Diana Franklin, and Margaret Martonosi. 2017. Programming languages and compiler design for realistic quantum hardware. *Nature* 549 (September 2017), 180–187.

[5] Alexei Colin, Emily Ruppel, and Brandon Lucia. 2018. A Reconfigurable Energy Storage Architecture for Energy-harvesting Devices. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*.

[6] Miyuru Dayarathna, Yonggang Wen, and Rui Fan. 2016. Data center energy consumption modeling: A survey. *IEEE Communications Surveys & Tutorials* 18, 1 (2016), 732–794.

[7] Eric Diller and Metin Sitti. 2013. Micro-Scale Mobile Robotics. *Found. Trends Robot* 2, 3 (Sept. 2013), 143–259. https://doi.org/10.1561/2300000023

[8] Jason Flinn. 2012. *Cyber Foraging: Bridging Mobile and Cloud Computing via Opportunistic Offload*. Morgan & Claypool Publishers.

[9] Kiryong Ha, Yoshihisa Abe, Tom Eiszler, Zhuo Chen, Wenlu Hu, Brandon Amos, Rohit Upadhyaya, Padmanabhan Pillai, and Mahadev Satyanarayanan. 2017. You Can Teach Elephants to Dance: Agile VM Handoff for Edge Computing. In *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*.

[10] John H. Howard, Michael L. Kazar, Sherri G. Menees, David A. Nichols, Mahadev Satyanarayanan, Robert N. Sidebotham, and Michael J. West. 1988. Scale and Performance in a Distributed File System. *ACM Transactions on Computer Systems* 6, 1 (1988).

[11] Y. Huang, N. Guo, M. Seok, Y. Tsividis, and S. Sethumadhavan. 2016. Evaluation of an Analog Accelerator for Linear Algebra. In *ACM/IEEE Intl. Symp. on Computer Architecture (ISCA)*.

[12] Johannes James, Vikram Iyer, Yogesh Chukewad, Shyamnath Gollakota, and Sawyer B. Fuller. 2018. Liftoff of a 190 mg Laser-Powered Aerial Vehicle: The Lightest Wireless Robot to Fly. In *Proceedings of the IEEE Int'l Conference on Robotics and Automation*. Brisbane, Australia.

[13] Craig D. LaBoda, Alvin R. Lebeck, and Chris L. Dwyer. 2017. An Optically Modulated Self-Assembled Resonance Energy Transfer Pass Gate. *Nano Letters* 17, 6 (2017), 3775–3781. https://doi.org/10.1021/acs.nanolett.7b01112 arXiv:https://doi.org/10.1021/acs.nanolett.7b01112 PMID: 28488874.

[14] Etienne Le Sueur and Gernot Heiser. 2010. Dynamic voltage and frequency scaling: The laws of diminishing returns. In *Proceedings of Int'l conference on Power aware computing and systems*.

[15] Vincent Liu, Aaron Parks, Vamsi Talla, Shyamnath Gollakota, David Wetherall, and Joshua R Smith. 2013. Ambient backscatter: wireless communication out of thin air. In *Proceedings of ACM SIGCOMM*, Vol. 43. 39–50.

[16] Xing Liu, Tianyu Chen, Feng Qian, Zhixiu Guo, Felix Xiaozhu Lin, Xiaofeng Wang, and Kai Chen. 2017. Characterizing smartwatch usage in the wild. In *Proceedings of ACM MobiSys*. 385–398.

[17] Brandon Lucia, Vignesh Balaji, Alexei Colin, Kiwan Maeng, and Emily Ruppel. 2017. Intermittent Computing: Challenges and Opportunities. In *Proceedings of the 2nd Summit on Advances in Programming Languages*.

[18] Brandon Lucia and Benjamin Ransford. 2015. A Simpler, Safer Programming and Execution Model for Intermittent Systems. In *Proc. of the 36th ACM SIGPLAN Conf. on Prog. Lang. Design and Implementation*. Portland, OR.

[19] Yunfei Ma, Zhihong Luo, Christoph Steiger, Giovanni Traverso, and Fadel Adib. 2018. Enabling deep-tissue networking for miniature medical devices. In *Proceedings of ACM SIGCOMM*. 417–431.

[20] Advait Madhavan, Timothy Sherwood, and Dmitri Strukov. 2016. Energy Efficient Computation with Asynchronous Races. In *Proc. of the 53rd Annual Design Automation Conf.* Austin, TX.

[21] Kiwan Maeng and Brandon Lucia. 2018. Adaptive Dynamic Checkpointing for Safe Efficient Intermittent Computing. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*.

[22] Chulhong Min, Youngki Lee, Chungkuk Yoo, Seungwoo Kang, Sangwon Choi, Pillsoon Park, Inseok Hwang, Younghyun Ju, Seungpyo Choi, and Junehwa Song. 2015. PowerForecaster: Predicting smartphone power impact of continuous sensing applications at pre-installation time. In *Proceedings of ACM SenSys*. 31–44.

[23] Sparsh Mittal and Jeffrey S Vetter. 2015. A survey of methods for analyzing and improving GPU energy efficiency. *Comput. Surveys* 47, 2 (2015).

[24] James H. Morris, Mahadev Satyanarayanan, Michael H. Conner, John H. Howard, David S. Rosenthal, and F. Donelson Smith. 1986. Andrew: A Distributed Personal Computing Environment. *Communications of the ACM* 29, 3 (1986).

[25] Brian D. Noble, M. Satyanarayanan, Dushyanth Narayanan, J. Eric Tilton, Jason Flinn, and Kevin R. Walker. 1997. Agile Application-Aware Adaptation for Mobility. In *Proc. of the 16th ACM Symp. on Operating Systems Principles*.

[26] Vijay Pillai, Harley Heinrich, David Dieska, Pavel V Nikitin, Rene Martinez, and KV Seshagiri Rao. 2007. An ultra-low-power long range battery/passive RFID tag for UHF and microwave bands with a current consumption of 700 nA at 1.5 V. *IEEE Transactions on Circuits and Systems* 54, 7 (2007), 1500–1512.

[27] Statista: That Statistics Portal. 2016. Projected size of the global market for RFID tags from 2016 to 2020. https://www.statista.com/statistics/299966/size-of-the-global-rfid-market/. (2016).

[28] Mahadev Satyanarayanan. 1990. Scalable, Secure, and Highly Available Distributed File Access. *IEEE Computer* 23, 5 (1990).

[29] Mahadev Satyanarayanan. 1996. Fundamental Challenges in Mobile Computing. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*.

[30] Mahadev Satyanarayanan. 2001. Pervasive Computing: Vision and Challenges. *IEEE Personal Communications* 8, 4 (2001).

[31] Mahadev Satyanarayanan. 2002. The Evolution of Coda. *ACM Transactions on Computer Systems* 20, 2 (2002).

[32] Mahadev Satyanarayanan. 2014. A Brief History of Cloud Offload. *ACM GetMobile* 18, 4 (2014).

[33] Mahadev Satyanarayanan. 2017. The Emergence of Edge Computing. *IEEE Computer* 50, 1 (2017).

[34] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Caceres, and Nigel Davies. 2009. The Case for VM-Based Cloudlets in Mobile Computing. *IEEE Pervasive Computing* 8, 4 (2009).

[35] Mahadev Satyanarayanan, John H. Howard, David A. Nichols, Robert N. Sidebotham, Alfred Z. Spector, and Michael J. West. 1985. The ITC Distributed File System: Principles and Design. In *Proceedings of the 10th ACM Symposium on Operating System Principles*.

[36] Eric R. Scerri. 2007. *The Periodic Table: Its Story and Its Significance*. Oxford University Press.

[37] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar. 2016. ISAAC: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars. In *ACM/IEEE 43rd Annual Intl. Symp. on Computer Architecture*.

[38] Dan Siewiorek, Asim Smailagic, and Thad Starner. 2008. *Application Design for Wearable Computing*. Morgan & Claypool Publishers.

[39] James E. Smith. 2014. Efficient Digital Neurons for Large Scale Cortical Architectures. In *Proc. of the 41st Annual Intl. Symp. on Computer Architecuture (ISCA '14)*. Minneapolis, MN.

[40] Andrew S. Tanenbaum, M. Frans Kaashoek, Robbert Van Renesse, and Henri E. Bal. 1991. The Amoeba Distributed Operating System - A Status Report. *Computer Communications* 14 (1991), 324–335.