FEATURE ARTICLE

The Duality of "Carry" and "Find" in Mobile and Pervasive Systems

Mahadev Satyanarayanan, *Carnegie Mellon University, Pittsburgh, PA, 15213, USA*

This article examines when a mobile system should view a resource as portable (and hence carried by its user), and when it can be viewed as pervasive (and hence easily found whenever and wherever needed). It shows that the anticipated worst case scenario while mobile is the critical determinant. If the resource is easily and reliably available at all visited locations, then it can be treated as a pervasive resource. If there is reasonable doubt about its availability or attributes at even one visited location, then it has to be treated as portable. Using this resource-centric viewpoint, this article highlights the unique role of the cloud in mobile computing, as well as its limitations. It discusses how some legacy systems can be rearchitected using modern technology, and how emerging real-time mobile AI systems will have to straddle the "carry" versus"find" divide.

In the inaugural issue of this publication, I characterized pervasive computing environments as "saturated with computing and communication yet gracefully integrated with human users."[1] A decade earlier, I had observed that portability or wearability is a severe constraint on mobile computing hardware: "Regardless of future technological advances, a mobile unit's weight, power, size, and ergonomics will render it less computationally capable than its static counterpart."[2]

The advances in mobile and pervasive computing in the 20–30 years since these early observations suggest that there is an inherent duality between the concepts of portability (mobile computing) and ubiquity (pervasive computing). Many end goals can be achieved using either approach. However, the resulting solutions will embody very different reliability, cost, and convenience tradeoffs, and lead to quite different business models. This duality is explored in the rest of this article.

The article is organized as follows. It starts by examining how the everyday resources shown in Figure 1 are used (see the "Usage Model of Some Common Resources" section). From those insights, it puts forth a design principle to help a designer choose between "carry" and "find" for any specific resource (see the "A Design Principle and its Economics" section). The article then examines how the Internet makes cloud-based resources ubiquitous, and motivates the need for disconnected operation (see the "Cloud + Internet = Ubiquity" section). End-to-end latency is identified as a key limitation of Internet-delivered cloud resources, motivating edge computing as a way to overcome this limitation (see "Latency: A Challenging Resource Attribute" section). Through the lens of "carry" versus "find," this article examines and reimagines a number of mobile and pervasive computing use cases (see the "Reimagining Cyber Resources" section). Emerging real-time mobile AI applications are shown to be particularly challenging, and only implementable through a combination of both metaphors. Finally, the "Conclusion" section concludes the article with a summary of its key points.

While this paper focuses on technical considerations, business considerations inevitably enter into the equation. A few examples are highlighted, but there are far too many to list in full. An important take-away message of this paper is that edge computing—via a tier of cloud-like resources that are in close network proximity to mobile devices—can be a powerful catalyst for advancing real-time mobile AI applications that have transformative value for society.

© 2026 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
Digital Object Identifier 10.1109/MPRV.2026.3676231

Resource	Portable (Carry)	Pervasive (Find)
Air		✓
Light	✓	✓
Water	✓	✓
Telephone (pre-2000)		✓
Telephone (now)	✓	✓

Size of check mark roughly indicates how extensively a resource is used via a portable (i.e., "carry") or pervasive (i.e., "find") paradigm.

FIGURE 1. Dominant usage model of critical resources.

Identifying business models that can sustain such a computing tier remains an major challenge today.

USAGE MODEL OF SOME COMMON RESOURCES

Consider the first resource in Figure 1: air. We cannot live for more than a few minutes without fresh air. We assume, without question, that it is available everywhere we go. In other words, our mobility is enabled by the ubiquity of this resource. The only times we treat air as a portable resource, rather than a ubiquitous one, is on rare and exotic occasions, such as when scuba diving, exploring outer space, or firefighting.

The second resource in Figure 1 is light. Humans are intensely visual creatures. Our survival depends on our visual skills. A billion years of evolution has resulted in our visual sensing and processing pipelines being exquisitely optimized for ambient lighting. In the past few hundred years, we have developed techniques to make light ubiquitous even at night. Today, we can easily illuminate rooms, homes, or even city blocks with easily controlled artificial lighting. We thus primarily treat light as a pervasive resource. We do have portable light sources, such as flashlights, but they rarely have to be used in a modern city. Portable light sources are needed only when infrastructure is absent (as on a camping trip), has suffered major failure (as in a natural disaster like a hurricane), or is unreliable (as in some developing countries).

Water is the third resource in Figure 1. Humans cannot survive for more than a few days without this resource. It is true that many people carry a water bottle to stay hydrated, but these are only caches—they rely on a water cooler or a faucet for refilling their caches. A hallmark of advancement from primitive conditions is the availability of a safe and reliable source of running water to all homes. In other words, it involves transforming water from a portable resource that has to be manually transported into a ubiquitous resource that is available on demand everywhere.

Modern civilization has made the telephone (fourth and fifth rows of Figure 1) an essential resource—it is difficult to function effectively today without it. What is unique about this resource is the metamorphosis that it has undergone over the last 25 years. Prior to about 2000, telephones were primarily landlines. An entire body of infrastructure and practices had evolved around wired telephony. Telephone booths with payphones were ubiquitous. When traveling, you could not be reached at a fixed phone number. For nearly a hundred years, telephones were firmly in the "Pervasive" class of resources. Yet, in just the past 25 years, they have flipped to "Portable." Cellular wireless technology enables you to reach a person on their phone without knowing their current location. That person, in turn, does not have to search for a payphone to make a call while traveling. The flipping of this resource is proof that barriers, such as deeply ingrained practices and heavy infrastructure investments can be overcome, if the end-user win from changing the doctrine of use is large enough. This flip involved massive change: wireless technologies, mobile phone hardware, and cell towers—a total revolution.

A DESIGN PRINCIPLE AND ITS ECONOMICS

A mobile system designer has to decide when a resource should be treated as portable, and when it should be treated as pervasive. In general, having to carry anything on one's travels is a nuisance: it may be large, heavy, and slow down progress; it can be misplaced, stolen, or damaged; the supply of the resource may run out. As the discussion of air in the "Usage Model of Some Common Resources" section exemplifies, mobility is hands-free and care-free at its best. Ubiquity trumps portability from the viewpoint of mobile user experience. This leads to the question: "Why should a resource ever be treated as portable rather than pervasive?"

To answer this question, it is useful to think through the process of packing for a long, multicity trip. What is the bare minimum that you need to pack? Your highly personalized items, such as clothing and prescription medications, certainly need to be packed, but easily found items, such as over-the-counter medications, can typically be purchased if needed. The same is true if you run out of toothpaste or razors. You do not have to carry and carefully monitor the use of a resource unless you are visiting a destination where

obtaining it may be difficult. This leads to the following design principle:

> *The anticipated worst case scenario while mobile determines whether a resource needs to be treated as portable or pervasive. If the resource is easily and reliably available at all visited locations, then it can be treated as a pervasive resource. If there is reasonable doubt about its availability or reliability at even one visited location, then it has to be treated as portable.*

Achieving reliable and plentiful availability of a resource everywhere typically implies significant economic investment and management effort. For example, ensuring reliable 5G wireless network coverage over a large area requires substantial capital investment in cell towers and related infrastructure, and in spectrum leases; it also requires substantial operational expenditure in maintenance and system administration. If underinvestment leads to spotty coverage in some areas, then you cannot totally rely on cloud services—support for transient disconnected operation then becomes essential. Treating a resource as pervasive requires a business model that supports the large cost of making that assumption true.

The economic burden of a portable resource falls entirely on its user. You buy your own laptop or smartphone. If improvements to a portable resource are clearly visible to a user (e.g., higher performance or bigger screen), then there is typically little resistance to a higher price for it. This is in contrast to a pervasive resource, such as the Internet or roads, whose cost is a shared burden on all its users. Improvements to a pervasive resource have to go through a complex amortization process before their per-user impact is clear. It may not be obvious to a user what hidden improvement is being paid for by a higher price. The economic model of a resource thus plays a critical role in how easily it can be incrementally deployed. The adoption model described in the classic "Crossing the Chasm"[3] only applies to new portable resources whose cost is borne by individual users. That model suggests that widespread dissemination of a new technology is likely if a critical group of "early majority" users can be convinced to adopt the technology. Unfortunately, the model does not translate easily to a pervasive resource whose improvement requires the consensus of a large body of individuals to incur a major expenditure. The transformation of a resource from portable to pervasive or vice versa thus involves significant nontechnical considerations.

There may also be unintended consequences of such a transformation. Today, the smartphone has become an intensely personal device. It has become so personal that possessing it is taken as proof of identity, via two-factor authentication. It is unlikely that the pioneers of cellular telephony foresaw this transformation. Their primary focus was the reachability of a user for voice communication, regardless of location. Today, it is fair to say that the most valuable roles of a smartphone are as an identification device, a wallet, a navigation device, and as an email/SMS device. These roles would not be possible with landlines. Voice communication, which inspired cellular telephony, is still valuable but hardly the dominant usage modality of a smartphone.

CLOUD + INTERNET = UBIQUITY

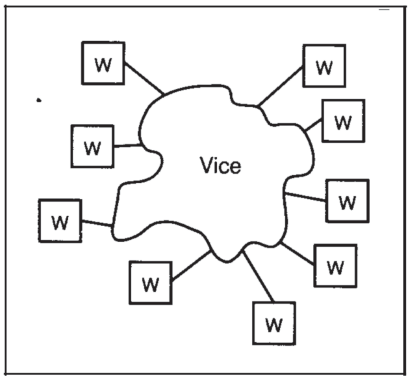
Why the Cloud is All-Powerful

Any cloud-based service can be viewed as a pervasive cyber resource. Consider *telepresence,* defined as "a technology that enables a person to perform actions in a distant or virtual location as if physically present in that location."[5] Just two decades ago, this was still science fiction. Today, millions of users worldwide achieve telepresence via Zoom, Teams, or Webex. Internet improvements, combined with a cloud-centric design, makes telepresence a ubiquitous cyber resource today.

A sufficiently scalable implementation of a cloud service at a single data center could, in theory, cover all its users anywhere in the world via the Internet. Such centralization offers extreme economies of scale, and drives down the cost of making that cyber resource ubiquitous. This is, by far, the lowest cost approach to offering a ubiquitous cyber resource. Unless attributes other than ubiquity matter, no other implementation strategy can hope to be cost-competitive with the fully centralized strategy of the cloud. Like Roman aqueducts that transformed water into a ubiquitous resource, the Internet makes cloud-based cyber resources ubiquitous.

A History Lesson for the Cloud

Alas, the phrase "Unless attributes other than ubiquity matter, ..." is a loaded one. Other attributes often *do* matter. One of the very first such attributes that emerged was *reliability*. We learned about the vulnerability of this attribute in a cloud context nearly 40 years ago, in the Andrew project.[6] This work achieved ubiquitous resource access at low management cost long before commercial cloud computing began (2005–2006). It can therefore be viewed as a technical



This figure originally appeared as Figure 1 of the 1990 retrospective paper on scalable, secure and highly available file access in AFS and its descendant, Coda [4]. The original caption read as follows:

> A high-level view of the Andrew architecture. The structure labeled "Vice" is a collection of trusted file servers and untrusted networks. The nodes labeled "W" are private or public workstations, or timesharing systems. Software in each such node makes the shared files in Vice appear as an integral part of that node's file system.

FIGURE 2. Private cloud circa 1990 (Source: the work of Satyanarayanan[4]).

forerunner of today's cloud-based services. In modern parlance, the mid-1980s Andrew project at Carnegie Mellon University (CMU) aimed to create a private cloud for the enterprise.

The very first cyber resource to be offered in Andrew was distributed file access—the ability to access your files as if they were local, no matter where you were on campus. The technical details of the Andrew File System (AFS) have been described in contemporary research papers.[7,8] Although the scope of ubiquity was initially campus-scale, it was later extended to the entire Internet.[9] The range of cyber services offered was also extended beyond file access.[10]

A 1990 retrospective paper on the AFS experience[4] summarized its lessons. Figure 2, reproduced verbatim from that paper, shows the vision of a cloud with the cyber service of file caching seamlessly integrated into the local file namespaces of edge nodes. This service made AFS files a ubiquitous resource. As the quoted text in Figure 3 shows, the power of ubiquity to enable mobility was well understood by 1990. The management simplification achieved by cloud centralization was also well understood. The system architecture in Figure 2 was first deployed for limited use on the CMU campus in 1985,[7] with more extensive deployment by November 1986.[6,8] User mobility via the "find" metaphor thus appeared well before its support via the "carry" metaphor—the NEC UltraLite[11] announced in October 1988, weighing less than 2 kg

> *User mobility is supported:* A user can walk to any workstation in the system and access any file in the namespace. 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 above text is a verbatim quote from the 1990 paper cited in Figure 2.

FIGURE 3. Mobility via ubiquity circa 1990 (Source: the work of Satyanarayanan[4]).

and running PC software, was the earliest portable computer that could plausibly claim that title.

The 1990 retrospective paper also provided the design rationale for a follow-on system called the Coda File System that could mask failures of the cloud. This rationale applies to virtually any cloud-based cyber resource today, and is best expressed by the opening sentences of the paper describing disconnected operation in Coda.[12]

> "Every serious user of a distributed system has faced situations where critical work has been impeded by a remote failure. His frustration is particularly acute when his workstation is powerful enough to be used standalone, but has been configured to be dependent on remote resources."

In other words, the more dependent you are on the cloud, the more painful its absence. As recently as October 2025, millions of users worldwide relearned this lesson when Amazon cloud services were disrupted for many hours.[13] Availability of cloud services is always a concern, but it is of particular urgency in "hostile" settings, such as military operations and in areas affected by natural disasters.[14]

Support for disconnected operation in Coda requires significant implementation complexity on clients.[12] A client has to be prepared for disconnection at any moment by ensuring that all critical files are already cached ("hoarding"). While disconnected, it has to log all updates and fake the presence of the cloud ("emulation"). When the cloud is reachable again, cache state has to be revalidated, and modifications propagated ("reintegration"). The latter step includes discovery and exception handling of situations where local updates during disconnection conflict with concurrent updates to the cloud. This irreducible complexity is the price that one pays for the ability to continue the illusion of resource ubiquity

while tolerating transient outages of the cloud. Today, the support for a Google Docs or Microsoft 365 object to be used offline requires this logic to be implemented in browser Javascript.

Independent of technical feasibility, business considerations may determine whether disconnected operation is worth supporting. Today, targeted advertising and cloud services are the dominant sources of revenue for many Internet businesses. There is little incentive for them to support disconnected operation since they derive no revenue for the complexity of providing that support. The user may be carrying hardware that is capable of disconnected operation, but its software "dumbs it down" to a device that is useless without the cloud.

Resource Dependence

The implementation complexity of disconnected operation is only necessary if the longest expected cloud outage exceeds the maximum tolerable period without a specific cyber resource. Going back to the discussion of basic resources in the "Usage Model of Some Common Resources" section, a camel can go weeks without water. Humans, on the other hand, can only survive for a few days. This reasoning leads to the following three alternatives for cloud-based mobile systems:

- ensure fallback strategies that provide the resource (possibly in degraded form) during cloud outages;
- ensure that cloud outages are never long;
- ensure a high tolerance level for resource deprivation.

Like a hybrid automobile that needs to seamlessly switch between electricity and gasoline, the first approach involves significant implementation complexity. The second approach is the implicit premise of today's cloud-based services. Although this premise may not be true in practice, the service provider is typically absolved of legal responsibility for the consequences through fine print in the service agreement. Further, even if the cloud service itself is operational, the Internet may be unreliable at the client's location. The third approach has significance for the attribute of latency, which is discussed in the next section.

LATENCY: A CHALLENGING RESOURCE ATTRIBUTE

Beyond reliability, one of the most difficult to control attributes of a cyber resource is its *end-to-end latency,* whose breakdown includes many unpredictable components in addition to the actual processing time on a server. The overheads include first-hop wireless network transmission to and from a mobile device, queuing and transmission delays at each wired network hop on the Internet, and queuing and scheduling delays due to multitenancy on the cloud server. We have learned to tame the last of these (multitenancy) from the lessons learned over 20 years of cloud computing. Unfortunately, the same is not true for connectivity from a mobile device to a cloud server.

Metric	Private CBRS 4G LTE Network to On-Prem Server	Commercial 5G T-Mobile Network to AWS-East
Measurements	1057	1013
Hop Count Mean	4.0	12.2
Hop Count Std. Dev.	0	1.4
Hop Count Max	4.0	15.0
Measurements	6246	5094
RTT Mean (ms)	22.0	37.6
RTT Std. Dev. (ms)	5.5	13.4
RTT p75 (ms)	23.0	46.0
RTT p95 (ms)	27.0	60.0

These measurements were obtained over a one-week period in April 2024 from a smartphone on the CMU campus in Pittsburgh to AWS-East, located in Ashburn, VA [15]. The speed-of-light-in-fiber delay (one way) between these two end points is ~2 ms. The on-prem server at CMU is LAN-connected to the 4G LTE CBRS access point.

FIGURE 4. Network connectivity to on-prem server versus cloud.

Today, this end-to-end path may involve a surprisingly large number of networking hops, each of which adds unpredictability. For example, Figure 4 shows the measured hop count and round-trip time (RTT) over a one-week period in 2024 from a smartphone via cellular network first hop to its closest cloud server. The table compares two offloading configurations: a 4G LTE CBRS private wireless network to an on-prem server, and T-Mobile's commercial 5G wireless network to the closest AWS cloud server. Despite significant effort by carriers, such as T-Mobile, to optimize network routes to cloud data centers, there is a glaring difference in the hop count between the two configurations. Even though the on-air performance of 4G LTE is poorer than that of 5G, the lower RTT to the on-prem server makes it the better choice for offloading.

For many real-time AI applications, it is not just the mean value of the end-to-end delay that matters but also the length of its tail. A few long delays may be overweighted in their impact (such as user experience) on a cyber–human or cyber-physical system. This is in contrast to today's web-based services that have evolved to tolerate long-tailed latency

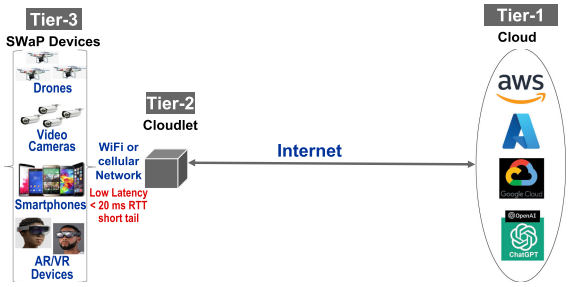


FIGURE 5. Tiered reference architecture.

distributions, with 99% tails sometimes stretching to many hundreds of milliseconds or even a few seconds.[16,17]

It is to reduce the magnitude and unpredictability of wide-area networking delays that cloudlet-based edge computing (see Figure 5) was originally proposed.[18,19] By introducing a new tier of computing (Tier-2 in Figure 5), the total end-to-end network path from mobile device to server is greatly shortened. Note that it is *network proximity* (measured in network hops) rather than physical proximity (measured in kilometers) that matters for terrestrial networking. At the speed of light in fiber, a millisecond translates to 200 km. Thus, it not physical distance but the multi-hop nature of the wide-area Internet that is at issue. Besides the latency benefit that originally motivated cloudlet-based edge computing, additional factors, such as data sovereignity and data privacy, have now emerged to further strengthen its value.[20,21] The implementation of Tier-2 in Figure 5 can take many forms. AWS Outpost, Azure Local, and Google Distributed Cloud are Tier-2 examples where the hardware is provided on-prem by the customer. In AWS Wavelength, the hardware is located within a telco's premises for wireless network customers. OpenClaw[22] is an open-source implementation of Tier-2 that is run on a user's own desktop, gaming laptop, etc., to provide low-latency access to large language models and other services. Regardless of the implementation, a critical requirement of Tier-2 is that the hardware does not suffer from the extreme constraints of mobility that are discussed in the next paragraph. It is thus closer in technical specifications to Tier-1 hardware than to Tier-3 hardware.

The obvious solution to reduce the latency of cloud-based cyber resources is to treat the resource as a portable resource, and to execute all its relevant code entirely on-device. Alas, the need for mobile devices to be small, lightweight, and portable makes this impossible for compute-intensive AI applications. The

	Typical Server		Typical Mobile 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 Explorer Edition	2.4 GHz (2 cores)
2016	Intel Xeon E5-2698v4	88.0 GHz (2x20 cores)	Samsung Galaxy S7	7.5 GHz (4 cores)
			Microsoft HoloLens	4.16 GHz (4 cores)
2020	Intel Xeon Gold 6348H	110.4 GHz (2x24 cores)	Samsung Galaxy S20	18.5 GHz (8 Cores)
			Microsoft HoloLens 2	19.0 GHz (8 cores)
			Google Glass Enterprise Ed. 2	15.2 GHz (8 cores)
2025	Intel Xeon 6732P	243.2 GHz (2x32 cores)	Pixel 9 Pro	18.6 GHz (8 Cores)
			OnePlus 13	29.8 GHz (8 cores)
			RayNeo X2	17.3 GHz (8 cores)

Adapted from our earlier work [23].
"Speed" = (number of processors × number of cores × clock speed)
Only CPU cores are included: GPUs and other accelerators are ignored.

FIGURE 6. Mobility penalty over three decades.

gap in the compute power of a state-of-the-art (SOTA) mobile device, and a SOTA server is referred to as its *mobility penalty*.[23] This irreducible performance gap can be viewed as the price paid for being portable or wearable. Figure 6 illustrates how the mobility penalty has stubbornly persisted over three decades. The order-of-magnitude gap between the second and third columns is the mobility penalty, which runs headlong into AI's ever-growing appetite for processing. Figure 7 shows the weight and processing power of NVIDIA's lightest GPUs. For comparison, the GPU on a SOTA

NVIDIA Jetson Type	Weight† (g)	INT8 TOPS (dense)	FP32 TFLOPS (dense)
Orin Nano 4GB	158	10	0.6
Orin Nano 8GB	176	33	1.3
Xavier NX	182	21	1.4
AGX Xavier	649	32	1.4
Orin NX 8GB	700	35	1.6
Orin NX 16GB	873	50	1.9
AGX Orin 64GB	873	138	5.3
AGX Thor T5000	1856	170	8.1
‡NVIDIA RTX 4090	2180		82.6

†Includes cooling such as heat sink, fan, etc. but not extra battery
‡Server GPU, shown for comparison
Jetson is NVIDIA's family of lightweight GPUs for Tier-3.

FIGURE 7. Weight and processing power of NVIDIA Jetson GPUs.

smartphone today (Google Pixel 9) is rated at 1.7 TFLOPS, placing it in the middle of the GPUs in Figure 7. The server-class GPU in the last row (NVIDIA RTX 4090) is 40–130 times as powerful as these low-end GPUs.

Recent work[15,24] shows that the mobility penalty in 2026 remains far too large for SOTA AI algorithms to be executed at real-time speeds (i.e., below 100 ms). While mobile devices will continue to improve, SOTA AI algorithms will continue to demand ever-more resources. For the near future (possibly for as long as a decade or two), the most accurate real-time AI services will have to be viewed as pervasive resources rather than portable ones. The capital and operational expense of creating and maintaining decentralized cloudlet infrastructure is the price that has to be paid for deploying accurate real-time AI at scale today.

REIMAGINING CYBER RESOURCES

The "carry versus find" lens is a valuable tool for examining implicit assumptions about user experience, resource availability, and business models. To summarize and recap, the following holds.

- *Carrying less improves mobility.* If mobility is the paramount design consideration, then it is preferable for a resource to be treated as pervasive rather than portable.
- *Ensuring that a resource is pervasive over an extensive coverage area can be expensive.* Identifying a business model that is capable of sustaining resource ubiquity becomes a crucial consideration. Treating a resource as portable typically leads to simpler business models that are more amenable to incremental deployment.
- *While cloud-based cyber resources are cheap to deploy Internet-wide, meeting their requirements of reliability and low end-to-end latency can be difficult.* Edge computing, realized as a tier of cloudlets that have network proximity to mobile devices, may be the only way to achieve low latency for SOTA AI resources today.

Building on these observations, and inspired by the the "Usage Model of Some Common Resources" section example of smartphones, we ask: *"Are any cyber resources ripe to flip from pervasive to portable, or vice versa?"* In the sections below, we explore the three cyber resources shown in Figure 8.

Resource	Portable	Pervasive
Smart Spaces (today)		✓
Smart Spaces (future)	??	
Mobile PC (today)	✓	
Mobile PC (future)		??
Real-Time AI (today)		✓
Real-Time AI (future)	??	

FIGURE 8. Some cyber resources.

Smart Spaces

The concept of a *smart space* was big in the early days of pervasive computing. For example, the Gator Tech Smart House of the early 2000s[25] was a widely cited example of how the integration of sensing and compute into the fabric of a building could make it safer for senior citizens. The inaugural issue of this publication described a similar effort.[26]

Unfortunately, the economics of smart spaces are challenging. It takes substantial capital investment by an organization to create a smart space that is richly equipped with sensors and actuators. The level of technology is frozen for some time after creation, since it is difficult and expensive to continuously track improvements. Justifying the investment cost is difficult, especially when the benefit is not easily quantified.

Smart whiteboards offer an excellent example of this economic tradeoff. Using special markers, you can write on these and have your writing immediately be recognized by the whiteboard. Rough sketches can be cleaned up in real time. The current contents can be captured before the board is wiped clean at the touch of a button. A different approach is to use a conventional whiteboard, and a smartphone to capture images of it. The image can be uploaded to the cloud for AI to recognize text, clean up drawings, etc. The experience is not quite as rich and seamless, but costs almost nothing. Most importantly, any conventional whiteboard becomes "smart."

More generally, viewing smart spaces as portable rather than pervasive changes the investment equation substantially. Augmented reality (AR) can be viewed as a technology that makes the space around you "smart." The ability to transform any space temporarily into a smart space is a powerful capability. A collection of colocated mobile users could bring a smart space into existence wherever they happen to be. Work done over a decade ago, showed how any flat surface could be made into a keyboard for input by projecting an image of the keyboard and sensing finger movement relative to the image.[27] About the same time, a prototype smartphone that could project an image of its screen on a wall was demonstrated.[28]

These nascent technologies have had a decade to mature now. By combining such technologies with the first-personal smartglass viewpoints of multiple users into a collective AR experience, it is easy to imagine smart spaces being created on demand wherever needed. Cloudlet-based real-time mobile AI (see the "Real-Time Mobile AI" section) obtained via "find" is a key enabler of this vision.

Mobile PC

In the mobile computing timeline, laptops are ancient technology that first appeared nearly 40 years ago. At a minimum weight of 1 kg today, laptops are far heavier than smartphones (roughly 200 g) that have to be carried anyway for two-factor authentication. Yet, there continues to be a persistent demand for laptops, topping 180 million units worldwide in 2025.[29] What drives this demand for heavy mobile devices? Could "find" replace "carry" for them?

The primary drivers of this demand are professional workers, such as architects, engineers, lawyers, creative artists, and software developers, who deliver their productive value through legacy software. They use authoring tools, such as the Microsoft Office Suite, Adobe Acrobat, Adobe Photoshop, GIMP, AutoDesk Revit, Google Earth, and software development tools such compilers and debuggers. Such precisely customized tools are unlikely to be available on generic hardware. Delivering them from the cloud is one possibility. There have been a few successful efforts (such as Microsoft 365) to rewrite the most popular of these authoring tools in a way that retains their user-facing and data-facing functionality, but leverages cloud computing. However, there are no clear business models for recouping the high cost of rewriting authoring tools that have much lower volumes of use.[30]

Rewriting can be avoided by encapsulating a tool and all of its dependencies (including its operating system) into a virtual machine (VM), and delivering it from the cloud to generic local hardware that is obtained using the "find" metaphor. The generic hardware can be used as a thin client of a VM instance that is executing in the cloud, or as a thick client that executes a local VM instance. This VM-based approach to making laptop/desktop environments ubiquitous was first demonstrated over two decades ago.[31,32]

The term *Virtual Desktop Infrastructure (VDI)* is used today for commercial implementations of thin client access to cloud-based VM execution. Unfortunately, the fine-grain, tightly interleaved *remote desktop protocol* interactions of VDI are acutely sensitive to the long-tailed latency of many-hop WANs.[30] Network proximity ensures that a cloudlet-based implementation of the VDI concept (*EdgeVDI*) does not suffer from this limitation[30]—the ubiquity of EdgeVDI can replace the portability of a laptop. Although the relevant business models and thin client products still remain to be developed, the technical feasibility of this vision is not in doubt.

EdgeVDI improves upon the ubiquity offered by caching in distributed file systems such as AFS and Coda (see the "A History Lesson for the Cloud" section). The latter only deliver files on demand, whereas EdgeVDI delivers the entire execution environment, including the precise version of the operating system and any customizations to it. In other words, the fidelity of the "find" abstraction offered by EdgeVDI is much closer to that of the "carry" ideal.

Real-Time Mobile AI

Until now, we have viewed "portable" versus "pervasive" as a binary choice. Looking ahead, there appears to be at least one emerging use case where "both" might be the optimal choice. As discussed in the "Latency: A Challenging Resource Attribute" section, implementing accurate real-time AI applications on mobile devices is an almost impossible design task today. On the one hand, the mobility penalty severely limits the accuracy and speed of on-device AI. On the other hand, the networking and multitenancy overheads of offloaded execution eats into the real-time budget. The problem gets worse with tighter upper bounds on real-time execution.

Multitier AI fusion can combine results from concurrent execution at two or more tiers to obtain average-case execution that is both fast and accurate. The speed part is obvious: executing a weak model on-device at Tier-3 incurs no network latency or bandwidth cost. The accuracy part comes from multiple sources: (a) Tier-2 or Tier-1 can run much bigger models that are more accurate; (b) Tier-2 or Tier-1 may have knowledge that is not available at Tier-3; this could arise from sensing from other Tier-3 devices, or from inferred or archived knowledge that cannot be exported to Tier-3 for many reasons (including security, classified nature of knowledge, etc.).

How exactly do we fuse these results that may arrive spaced widely apart? This is achieved through a programmming abstraction called a *Correctable Operation (C-Op)*. The abstraction is best understood via a simple example. In a real-time setting on a mobile device, acting immediately upon a Tier-3 AI result may be valuable. Waiting for a better result from Tier-2 or

Tier-1 may result in mission failure. For example, if Tier-3 object detection indicates a target, then a drone may wish to start its descent to lower altitude for better sensing and confirmation right away. Waiting for confirmation from Tier-2 or Tier-1 before starting the descent may allow the target to escape. A C-Op enables speculative actuation that can partially mask the offload latency. If a more accurate result from another tier indicates that no target is present, then an exception is triggered to cancel the drone's descent and return it to its original location. A C-Op may include a lateness limit, beyond which no exception is triggered because the new results are too late to be useful. In effect, C-Ops leverage asymmetry in the cost-benefit analysis of correctness versus timeliness. They improve timeliness significantly if the Tier-3 result is correct, but only have a mild negative impact if it is proven to be wrong by Tier-2 or Tier-1. These concepts are explored more deeply in recent work.[33]

This approach to multitier AI fusion has a builtin mechanism for transient disconnected operation: the Tier-3 result is simply used without correction during network or cloud failures. A valuable side-effect of this approach its software engineering benefit. The code path for disconnected operation is usually difficult to debug because it is an exception handling mechanism. With a C-Op, this is the most frequently executed code path—correction is the exception. With slight modification, this approach also provides a natural tuning knob to balance speed and accuracy. The description above assumes that a Tier-3 result is used as soon as it is available. It would be simple to delay use of the result for a time, T. A larger value of T biases fusion toward accuracy, because there is a greater chance of receiving a result from a more accurate tier.

As we look to the future, it is useful to reflect upon the themes of "carry" and "find" in the context of real-time mobile AI. The need for user interaction and the need to support transient disconnected operation both suggest that a minimal amount of hardware has to be carried. Even using an extreme thin client model, a user at least needs to carry enough hardware to sense input, offload processing over a wireless network, and render output. In 2026, this can be achieved with commercial off-the-shelf smartwatches and smartglasses weighing 30–60 g.[34,35] By fusing AI from such Tier-3 devices with Tier-2 AI from cloudlets that are discovered via a "find" mechanism, such as Sinfonia,[36] accurate real-time mobile AI is achievable today. The new applications that are created through this approach can be evolved over time to leverage custom ASICs that eliminate the need for Tier-2.[15,23] From that point forward, a pure "carry" strategy will suffice for those applications.

Concept	Mobile Systems	IoT Systems
"Carry"	Device carried by user	Data/models pre-provisioned on system
"Find"	Obtain as needed from cloudlet or cloud	Obtain as needed from cloudlet or cloud
Resources	Consumed by user actions	Consumed by IoT mission
Disconnected Operation	Masks user pain	Improves mission success

FIGURE 9. Concept mapping to IoT.

Internet of Things (IoT)

This article has focused on the "carry" versus "find" duality of human-centric mobile systems. Today, the IoT is a driver and consumer of many of the sensing, device/edge/cloud processing, and wireless networking technologies discussed in earlier sections. Although IoT systems are not human-centric, the observations of this article have relevance to them. With the remapping of concepts shown in Figure 9, the "carry" versus "find" distinction can be applied to IoT systems. Ubiquity, latency, edge computing, scalability, reliability, disconnected operation, and other topics that have been discussed in previous sections have relevance to IoT systems as well. For example, an ultralight drone that can only carry a tiny payload faces the same mobility penalty for real-time onboard AI as a smartglass for wearable cognitive assistance.[37,38]

CONCLUSION

In his book "Mind Children," roboticist Hans Moravec observes that it was mobility that made intelligence valuable in biological evolution.[39] Trees do not need to be smart: they do not move, and hence do not get themselves into much trouble. In the animal kingdom, the ability to forage for critical resources, such as food and water, was greatly improved by intelligence. By its very nature, mobility imposes severe weight, size, and energy-efficiency requirements on hardware. Today, mobile computing is an integral part of our life. Its holy

grail is the seamless extension of human intelligence that was envisaged by Weiser.[40]

> "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."

How to seamlessly augment human intelligence using mobile hardware that is severely constrained by weight, size, and energy efficiency is an unsolved problem today. A key message of this article is that edge computing—introducing a tier of cloud-like computing resources that are in close network proximity to users with wearable devices—can be a powerful catalyst for advancing toward Weiser's vision.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under Grant CNS-2106862.

The content of the information does not necessarily reflect the position or the policy of the government and no official endorsement should be inferred. This work was done in the CMU Living Edge Lab, which is supported by Intel, Arm, Vodafone, Deutsche Telekom, CableLabs, Crown Castle, InterDigital, Seagate, Microsoft, the VMware University Research Fund, IAI, and the Conklin Kistler family fund. Any opinions, findings, conclusions or recommendations expressed in this document are those of the authors and do not necessarily reflect the view(s) of their employers or the above funding sources.

Many colleagues worldwide reviewed and helped to greatly improve an early draft of this paper: Rajesh Balan (SMU), Suman Banerjee (Wisconsin), Maria Ebling (West Point), Gil Goldman (CMU), Sumi Helal (Bologna), Fahim Kawsar (Glasgow), Bobby Klatzky (CMU), Marc Langheinrich (USI), Grace Lewis (CMU-SEI), Kent Monas (US Army), Jacob Scriffiny (CMU), Junjue Wang (Apple), Roy Want (Google), Lin Zhong (Yale). I am deeply indebted to all of them for their generosity of time and for their insights, thoughtful comments, and excellent suggestions for improving the paper.

REFERENCES

1. M. Satyanarayanan, "A catalyst for mobile and ubiquitous computing," *IEEE Pervasive Comput.*, vol. 1, no. 1, pp. 2–5, Jan.–Mar. 2002.
2. M. Satyanarayanan, "Hot Topics: Mobile computing," *IEEE Comput.*, vol. 26, no. 9, pp. 81–82, 1993.
3. G. Moore, *Crossing the Chasm*. New York, NY, USA: Harpercollins, 1991.
4. M. Satyanarayanan, "Scalable, secure and highly available distributed file access," *IEEE Comput.*, vol. 23, no. 5, pp. 9–18, May 1990.
5. Merriam-Webster, "Unabridged dictionary." Accessed: Dec. 4, 2025. https://unabridged.merriam-webster.com/
6. J. H. Morris, M. Satyanarayanan, M. H. Conner, J. H. Howard, D. S. Rosenthal, and D. F. Smith, "Andrew: A distributed personal computing environment," *Commun. ACM*, vol. 29, no. 3, pp. 184–201, Mar. 1986.
7. M. Satyanarayanan, J. H. Howard, D. N. Nichols, R. N. Sidebotham, A. Z. Spector, and M. J. West, "The ITC distributed file system: Principles and design," in *Proc. 10th ACM Symp. Operating Syst. Princ.*, Dec. 1985, pp. 35–50.
8. J. H. Howard et al., "Scale and performance in a distributed file system," *ACM Trans. Comput. Syst.*, vol. 6, no. 1, pp. 51–81, Feb. 1988.
9. M. Spasojevic and M. Satyanarayanan, "An empirical study of a wide-area distributed file system," *ACM Trans. Comput. Syst.*, vol. 14, no. 2, pp. 200–222, May 1996.
10. J. Rosenberg, C. Everhart, and N. S. Borenstein, "An overview of the andrew message system," in *Proc. ACM Workshop Front. Comput. Commun. Technol.*, Aug. 1987, pp. 99–108, doi: 10.1145/55483.55494.
11. Wikipedia, "NEC UltrLite." Accessed: Mar. 3, 2026. [Online]. Available: https://en.wikipedia.org/wiki/NEC_UltraLite
12. J. J. Kistler and M. Satyanarayanan, "Disconnected operation in the coda file system," *ACM Trans. Comput. Syst.*, vol. 10, no. 1, pp. 3–25, Feb. 1992.
13. R. McMillan, B. Lin, and S. McClain, "The day amazon broke the internet for millions of americans," *Wall Street J.*, Oct. 2025. [Online]. Available: https://www.wsj.com/tech/aws-outage-impact-what-happened-becce0e7
14. M. Satyanarayanan, G. Lewis, E. Morris, S. Simanta, J. Boleng, and K. Ha, "The role of cloudlets in hostile environments," *IEEE Pervasive Comput.*, vol. 12, no. 4, pp. 40–49, Oct.–Dec. 2013.
15. M. Satyanarayanan, Q. Dong, J. Xu, and P. Pillai, "Towards mobile AI that is fast and accurate," in *Proc. 7th Int. Conf. Cogn. Mach. Intell.*, Pittsburgh, PA, Nov. 2025, pp. 425–432.
16. K. Ha et al., "The impact of mobile multimedia applications on data center consolidation," in *Proc. IEEE Int. Conf. Cloud Eng.*, San Francisco, CA, Mar. 2013, pp. 166–176.

17. W. Hu et al., "Quantifying the impact of edge computing on mobile applications," in *Proc. 7th ACM SIGOPS Asia-Pacific Workshop Syst.*, Hong Kong, China, 2016, pp. 1–8.
18. M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-Based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct.–Dec. 2009.
19. M. Satyanarayanan, "The emergence of edge computing," *IEEE Comput.*, vol. 50, no. 1, pp. 30–39, Jan. 2017.
20. M. Satyanarayanan and N. Davies, "Augmenting cognition through edge computing," *IEEE Comput.*, vol. 52, no. 7, pp. 37–46, Jul. 2019.
21. N. Davies, N. Taft, M. Satyanarayanan, S. Clinch, and B. Amos, "Privacy mediators: Helping IoT cross the chasm," in *Proc. ACM HotMobile 2016*, Feb. 2016, pp. 39–44.
22. "OpenClaw: The AI That Actually Does Things," Accessed: Mar. 12, 2026. [Online]. Available: https://openclaw.ai
23. M. Satyanarayanan, N. Beckmann, G. A. Lewis, and B. Lucia, "The role of edge offload for hardware-accelerated mobile devices," in *Proc. 22nd Int. Workshop Mobile Comput. Syst. Appl.*, 2021, pp. 22–29.
24. Q. Dong, J. Xu, P. Pillai, and M. Satyanarayanan, "Does accurate real-time AI need edge offload?," in *Proc. 10th ACM/IEEE Symp. Edge Comput.*, Arlington, VA, Dec. 2025, pp. 1–18, doi: 10.1145/3769102.3770616.
25. S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, and E. Jansen, "The Gator Tech smart house: A programmable pervasive space," *IEEE Comput.*, vol. 38, no. 3, pp. 50–60, Mar. 2005.
26. V. Stanford, "Using pervasive computing to deliver elder care," *IEEE Pervasive Comput.*, vol. 1, no. 1, pp. 10–13, Jan.–Mar. 2002.
27. J. Wang, K. Zhao, X. Zhang, and C. Peng, "Ubiquitous keyboard for small mobile devices: Harnessing multipath fading for fine-grained keystroke localization," in *Proc. 12th Annu. Int. Conf. Mobile Syst., Appl., Serv.*, 2014, pp. 14–27.
28. "Lenovo Tech World–Smart Cast," 2015. [Online]. Available: https://www.youtube.com/watch?v=EEO1E6E9iQw
29. Evertiq, "Global notebook shipments expected to grow by 4.9% in 2025," Nov. 2024. Accessed: Dec. 14, 2025. [Online]. Available: https://evertiq.com/news/56827#
30. M. Satyanarayanan, T. Eiszler, J. Harkes, H. Turki, and Z. Feng, "Edge computing for legacy applications," *IEEE Pervasive Comput.*, vol. 19, no. 4, pp. 19–28, Oct.–Dec. 2020.
31. M. Kozuch and M. Satyanarayanan, "Internet suspend/resume," in *Proc. 4th IEEE Workshop Mobile Comput. Syst. Appl.*, 2002, pp. 40–46.
32. M. Satyanarayanan et al., "Pervasive personal computing in an internet suspend/resume system," *IEEE Internet Comput.*, vol. 11, no. 2, pp. 16–25, Mar./Apr. 2007.
33. Q. Dong, J. Xu, P. Pillai, and M. Satyanarayanan, "Real-Time Fusion of Device, Cloudlet, and Cloud AI," 2026, In review. [Online]. Available: https://openclaw.ai
34. SonoVax, "Sleek, Smart, and Packed With Features." Accessed: Mar. 7, 2026. [Online]. Available: https://sonovaxco.com/products/sleek-smart-and-packed-with-features
35. Samsung, "Galaxy Watch8." Accessed: Mar. 7, 2026. [Online]. Available: https://www.samsung.com/us/watches/galaxywatch8
36. M. Satyanarayanan et al., "Sinfonia: Cross-tier orchestration for edge-native applications," *Front. Internet Things*, vol. 1, 2022, Art. no. 1025247, doi: 10.3389/friot.2022.1025247.
37. M. Bala et al., "Democratizing drone autonomy via edge computing," in *Proc. 2023 IEEE/ACM Symp. Edge Comput.*, 2023, pp. 40–52.
38. M. Bala et al., "The OODA loop of cloudlet-based autonomous drones," in *Proc. 2024 IEEE/ACM Symp. Edge Comput.*, 2024, pp. 178–190.
39. H. Moravec, *Mind Children*. Cambridge, MA, USA: Harvard Univ. Press, 1988.
40. M. Weiser, "The computer for the 21st century," *Sci. Amer.*, vol. 265, pp. 94–105, Sep. 1991.

MAHADEV SATYANARAYANAN is the Jaime Carbonell University Professor of Computer Science at Carnegie Mellon University, Pittsburgh, PA, 15213, USA. His research interests include the challenges of performance, scalability availability and trust in information systems that reach from the cloud to the mobile edge of the Internet. Satyanarayanan received his Ph.D. degree in computer science from Carnegie Mellon University. He is a fellow of ACM, life fellow of IEEE, and a member of the National Academy of Engineering. Contact him at satya@cs.cmu.edu.