Resource-Driven Dynamic Adaptation

15-821 / 18-843
Fall 2024

Mahadev Satyanarayanan
School of Computer Science
Carnegie Mellon University

Recap: Constraints of Mobility

Resource poverty

- **vs. static elements of same era**
- **weight, power, size constraints**

Communication uncertainty

- **bandwidth / latency variation**
- **intermittent connectivity**
- **may cost real money**

Finite energy source

- **actions may be slowed or deferred**
- **communication costs energy**

Multi-modal Interaction

- **hands and eyes occupied**
- **speech/gesture recognition**
- **augmented reality**

Scarce user attention

- **focus of attention elsewhere**
- **lower human performance**
- **higher error rate**

Less security & robustness

- **theft, destruction more likely**
- **greater exposure to subversion**

Contradictory System Requirements

Favoring reliance on the cloud

- resource-poor clients
- poorer security & robustness of clients
- need for up-to-date information

If network quality didn't matter you would only need thin clients

Favoring standalone ability

- you may not be able to contact cloud
- communication may be expensive
  (real dollars, energy)

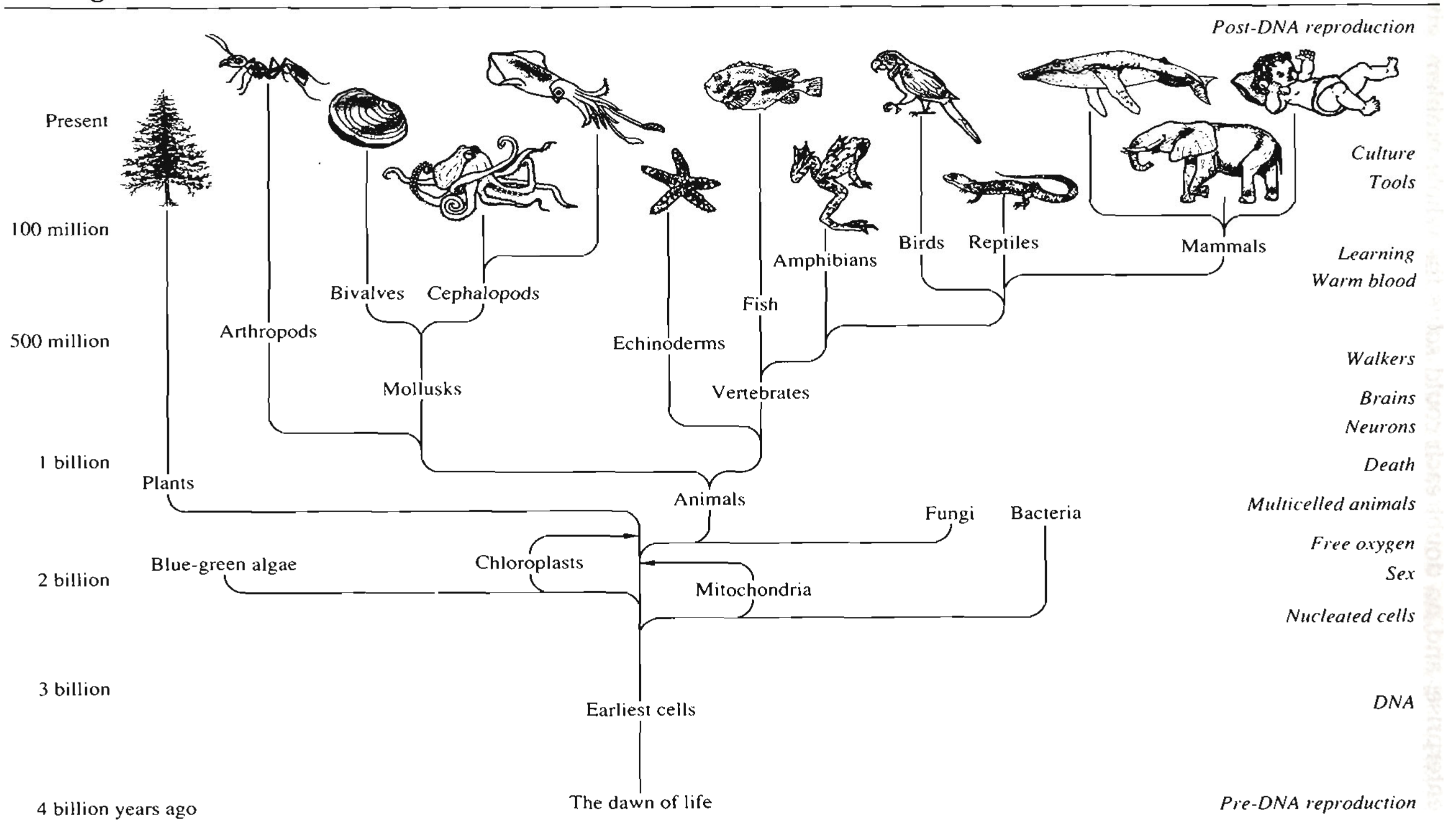
Disconnected and weakly-connected operation

Hence mobile systems must be *adaptive*

- rely on cloud when possible
- function autonomously if needed
- monitor and adjust to current conditions

morph thickness dynamically

Intelligence on Earth



From *"Mind Children"* by Hans Moravec

"This partial family tree of terrestrial organisms suggests the linkage between mobility and intelligence. One and a half billion years ago our unicelled ancestors parted genetic company with the plants. As single cells, both lines were free swimmers, but when plants became multicellular they specialized in being sedentary collectors of solar energy. Our animal forebears, on the other hand, remained ambulatory, the better to eat the plants and each other. While plants are enormously successful – the bulk of the earth's biosphere is vegetation, and the largest, most numerous, and longest-lived organisms are plants – **they show very little evolutionary tendency toward anything we would recognize as intelligence.** … "

Adaptation as a Selective Force

Mind Children

Hans Moravec (founder of SeeGrid, Inc)

Harvard University Press, Cambridge, MA, 1988.

First to observe linkage in evolution between

- mobility, and
- *emergence of intelligence as a selective force.*

Moravec's observations are about mobile robots

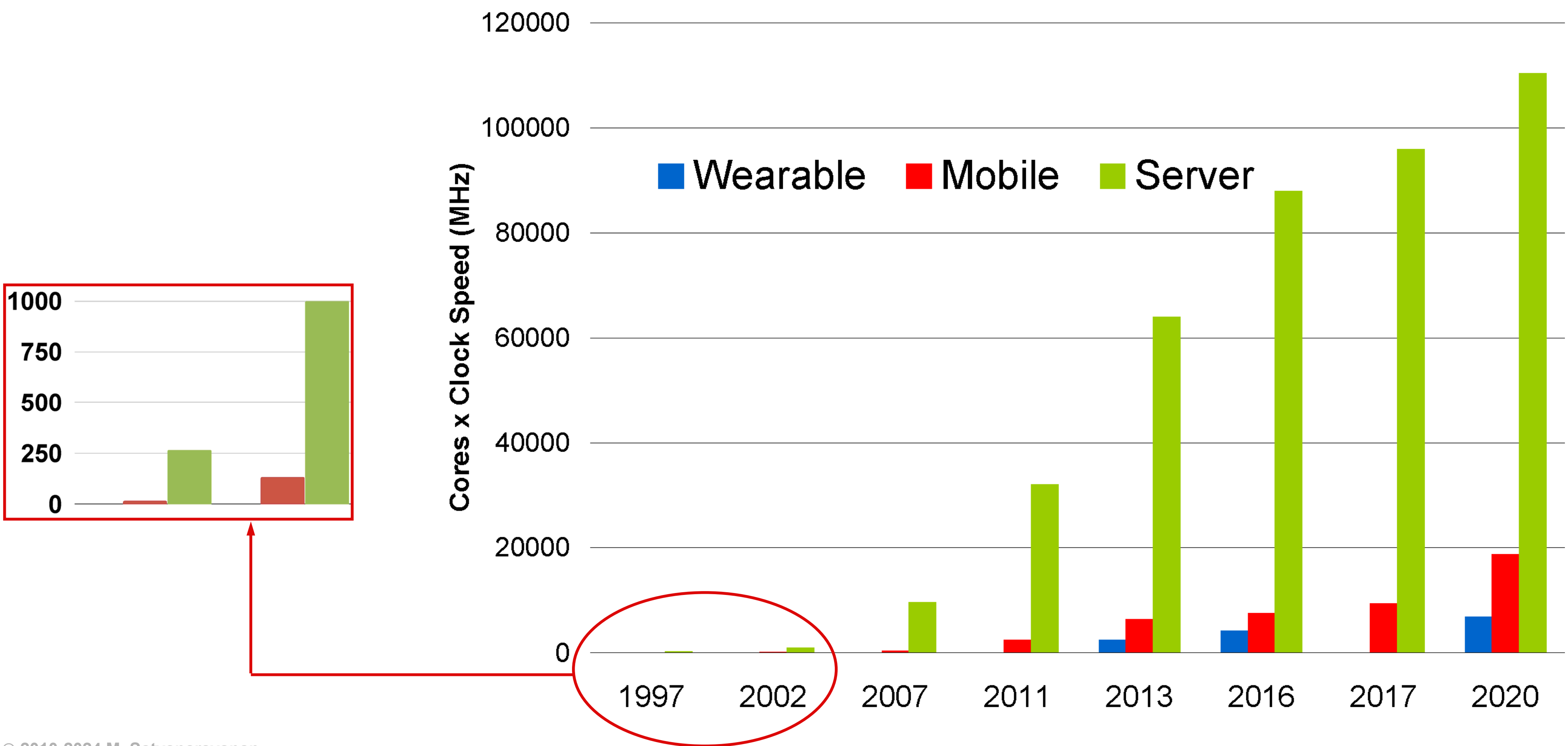
- most of their energy goes to locomotion
  being clever (higher level intelligence) is irrelevant to this part
- mobile computing focuses only on non-locomotive aspects
  importance of "intelligence" (aka adaptation) even higher

Offloading

(not available to nature!)

Mobility Penalty Over 25 Years

The *"mobility penalty"* is very real and persistent



Dilemma

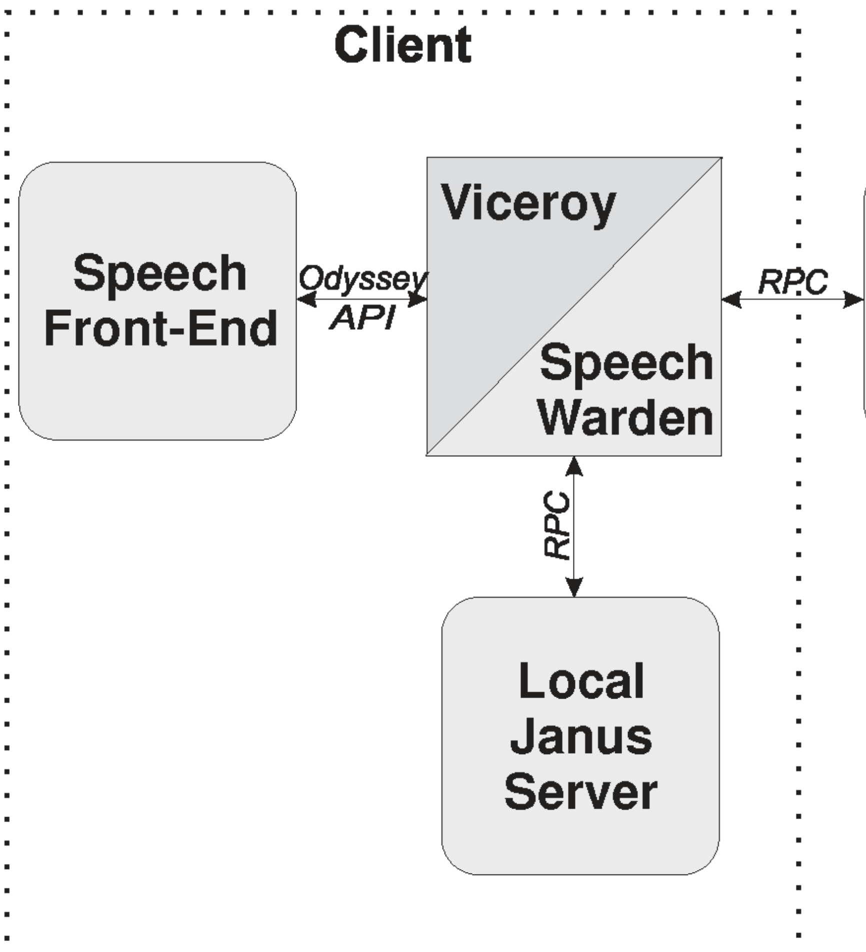
Our expectations are set by desktops / servers / cloud

But we want this anywhere, anytime, hands-free

(i.e., on mobile/wearable devices, while meeting thermal and energy constraints)

How do we square this circle?

Offload Computation from Device to Server



October 1997 (10 years before iPhone and Siri)

"Agile-Application Aware Adaptation for Mobility"
Noble, B., Satyanarayanan, M., Narayanan, D., Tilton, E., Flinn, J., Walker, K.
Proceedings of the 16th ACM Symposium on Operating Systems Principles, St. Malo, France, October 1997

Generalized as a Principle in 2001

Pervasive Computing: Vision and Challenges

M. SATYANARAYANAN, CARNEGIE MELLON UNIVERSITY

IEE Personal Communications, August 2001

2001-2020: huge amount of follow-on work by Satya's students and other researchers under the name "cyber foraging" or "compute offload"

Cyber Foraging

The need to make mobile devices smaller, lighter, and have longer battery life means that their computing capabilities have to be compromised. But meeting the ever-growing expectations of mobile users may require computing and data manipulation capabilities well beyond those of a lightweight mobile computer with long battery life. Reconciling these contradictory requirements is difficult.

Cyber foraging, construed as "living off the land," may be an effective way to deal with this problem. The idea is to dynamically augment the computing resources of a wireless mobile computer by exploiting wired hardware infrastructure. As computing becomes cheaper and more plentiful, it makes economic sense to "waste" computing resources to improve user experience. Desktop computers at discount stores already sell today for a few hundred dollars, with prices continuing to drop. In the forseeable future, we envision public spaces such as airport lounges and coffee shops being equipped with compute servers or data staging servers for the benefit of customers, much as table lamps are today. These will be connected to the wired Internet through high-bandwidth networks. When hardware in the wired infrastructure plays this role, we call it a *surrogate* of the mobile computer it is temporarily assisting.

We envision a typical scenario as follows. When a mobile computer enters a neighborhood, it first detects the presence of potential surrogates and negotiates their use. Communication with a surrogate is via short-range wireless peer-to-peer technolo-

Simplifying Cyber Foraging

Rajesh Krishna Balan

May 2006

CMU-CS-06-120

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

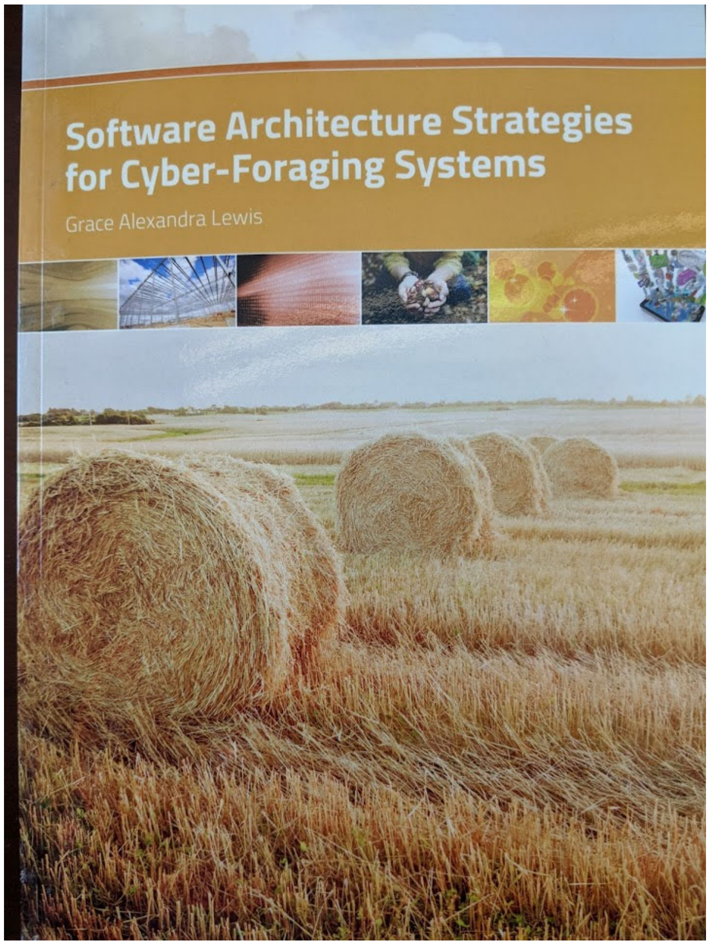
*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

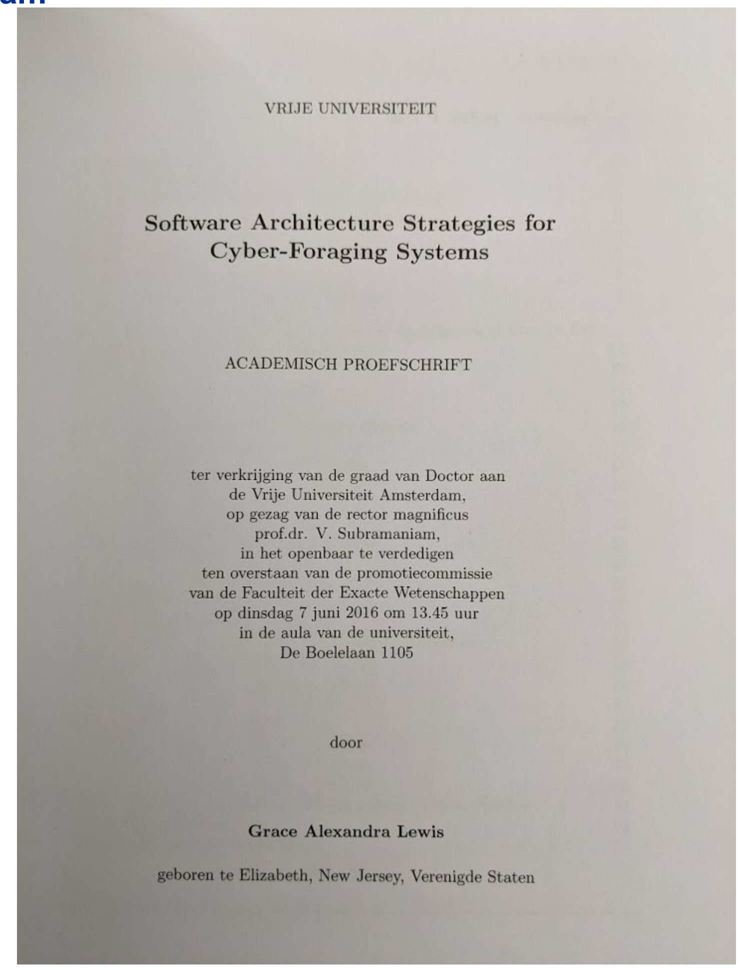
Thesis Committee:
Mahadev Satyanarayanan, Chair
David Garlan
Gregory Ganger
Srinivasan Seshan
Hari Balakrishnan, *Massachusetts Institute of Technology*

This research was partially supported by the National Science Foundation (NSF) under grant numbers ANI-0081396 and CCR-0205266, by a USENIX graduate fellowship, by an IBM Graduate Fellowship, and by an equipment grant from the Hewlett-Packard Corporation (HP). Any opinions, findings, and conclusions or

Grace Lewis PhD Thesis

2016 Vrije Universiteit, Amsterdam





VRIJE UNIVERSITEIT

Software Architecture Strategies for
Cyber-Foraging Systems

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van Doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. V. Subramaniam,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de Faculteit der Exacte Wetenschappen
op dinsdag 7 juni 2016 om 13.45 uur
in de aula van de universiteit,
De Boelelaan 1105

door

Grace Alexandra Lewis

geboren te Elizabeth, New Jersey, Verenigde Staten

Many non-CMU Efforts Too

MobiSys 2010

MAUI: Making Smartphones Last Longer with Code Offload

Eduardo Cuervo[†], Aruna Balasubramanian[‡], Dae-ki Cho[*], Alec Wolman[§], Stefan Saroiu[§], Ranveer Chandra[§], Paramvir Bahl[§]
[†]Duke University, [‡]University of Massachusetts Amherst, [*]UCLA, [§]Microsoft Research

OSDI 2012

OSDI 12 OCTOBER 8—10, 2012 HOLLYWOOD, CA
sponsored by USENIX in cooperation with ACM SIGOPS
usenix THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

COMET: Code Offload by Migrating Execution Transparently

connect with us

Authors:
Mark S. Gordon, D. Anoushe Jamshidi, Scott Mahlke, and Z. Morley Mao, *University of Michigan;* Xu Chen, *AT&T Labs —Research*

Aarhus University, Denmark, 2010

AARHUS UNIVERSITET

› Empowering Mobile Devices Through Cyber Foraging

Empowering Mobile Devices Through Cyber Foraging: The Development of Scavenger, an Open, Mobile Cyber Foraging System

Research output: Book/anthology/dissertation/report › Ph.D. thesis

- Mads Darø Kristensen, Denmark
- Department of Computer Science

Original language	English
Publisher	Department of Computer Science, Aarhus University
Number of pages	242
Publication status	Published - 2010

“Cyber Foraging” is more commonly known as “offloading” today

Optional Reading

Satyanarayanan, M.
"A Brief History of Cloud Offload"
GetMobile, Volume 18, Issue 4, October 2014

Where to Offload?

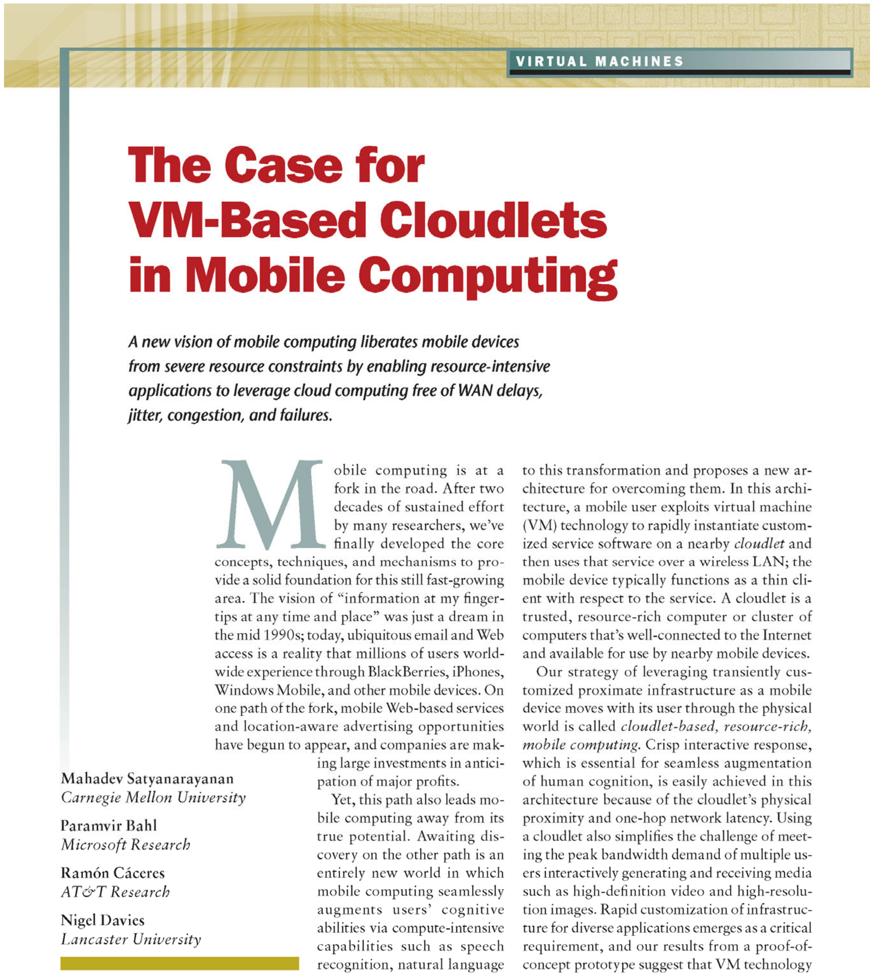
The Cloud Emerges (~2006-2010)
(along with exascale data centers & CDNs)

***Consolidation*, *Economies of Scale*, and *OpEx for Capex* are key themes**

At what price?

The price is *end-to-end latency of offload*

Edge Computing: Low-Latency Offload



VIRTUAL MACHINES

The Case for VM-Based Cloudlets in Mobile Computing

A new vision of mobile computing liberates mobile devices from severe resource constraints by enabling resource-intensive applications to leverage cloud computing free of WAN delays, jitter, congestion, and failures.

Mahadev Satyanarayanan
Carnegie Mellon University

Paramvir Bahl
Microsoft Research

Ramón Cáceres
AT&T Research

Nigel Davies
Lancaster University

Mobile computing is at a fork in the road. After two decades of sustained effort by many researchers, we've finally developed the core concepts, techniques, and mechanisms to provide a solid foundation for this still fast-growing area. The vision of "information at my fingertips at any time and place" was just a dream in the mid 1990s; today, ubiquitous email and Web access is a reality that millions of users worldwide experience through BlackBerries, iPhones, Windows Mobile, and other mobile devices. On one path of the fork, mobile Web-based services and location-aware advertising opportunities have begun to appear, and companies are making large investments in anticipation of major profits.

Yet, this path also leads mobile computing away from its true potential. Awaiting discovery on the other path is an entirely new world in which mobile computing seamlessly augments users' cognitive abilities via compute-intensive capabilities such as speech recognition, natural language

to this transformation and proposes a new architecture for overcoming them. In this architecture, a mobile user exploits virtual machine (VM) technology to rapidly instantiate customized service software on a nearby *cloudlet* and then uses that service over a wireless LAN; the mobile device typically functions as a thin client with respect to the service. A cloudlet is a trusted, resource-rich computer or cluster of computers that's well-connected to the Internet and available for use by nearby mobile devices.

Our strategy of leveraging transiently customized proximate infrastructure as a mobile device moves with its user through the physical world is called *cloudlet-based, resource-rich, mobile computing*. Crisp interactive response, which is essential for seamless augmentation of human cognition, is easily achieved in this architecture because of the cloudlet's physical proximity and one-hop network latency. Using a cloudlet also simplifies the challenge of meeting the peak bandwidth demand of multiple users interactively generating and receiving media such as high-definition video and high-resolution images. Rapid customization of infrastructure for diverse applications emerges as a critical requirement, and our results from a proof-of-concept prototype suggest that VM technology

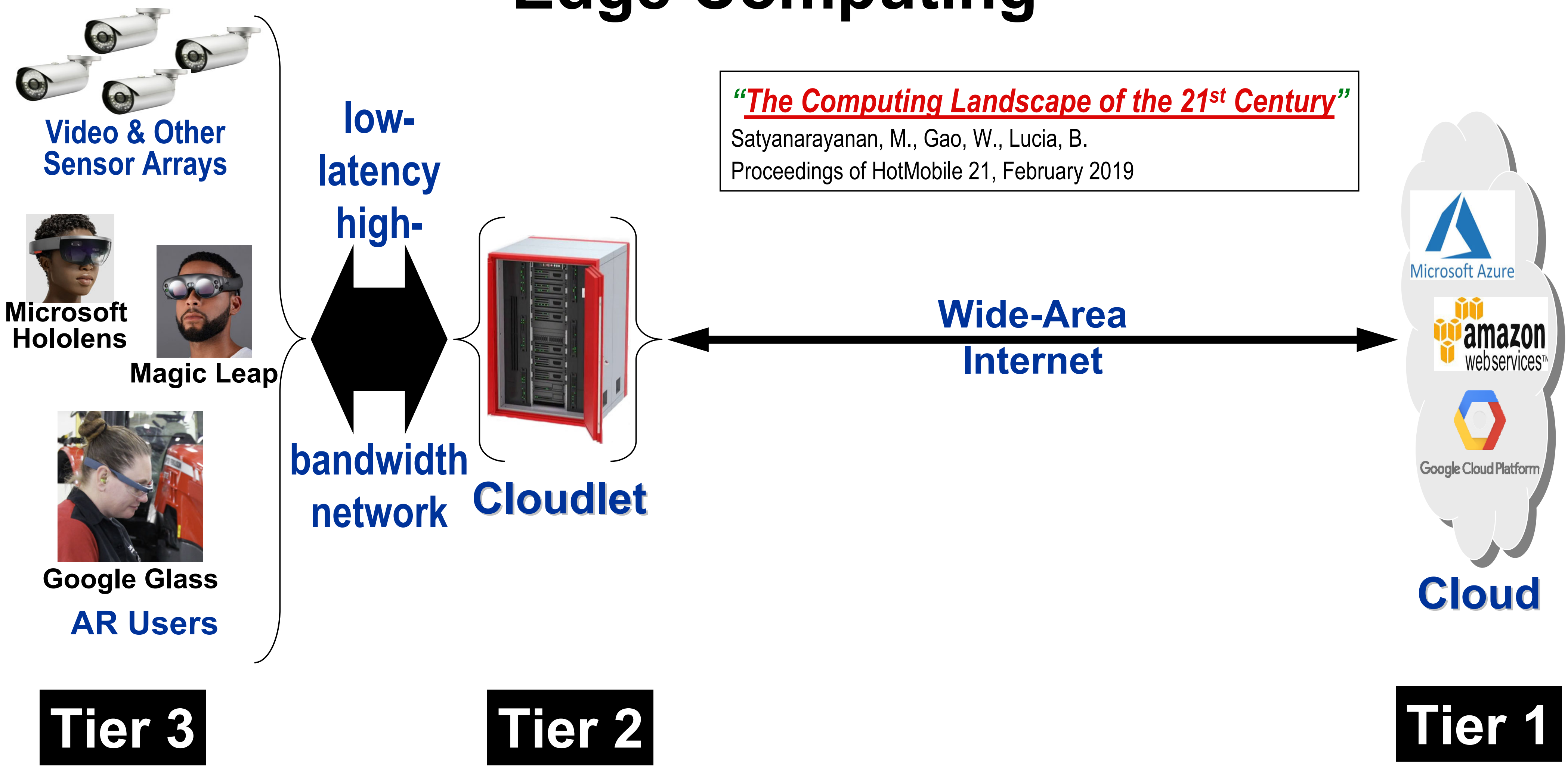
Published in October 2009

CMU, Microsoft, AT&T, Intel, Lancaster Univ authors

Now seen as the "founding manifesto" of Edge Computing

ACM SIGMOBILE 2022 Test of Time Award

Edge Computing



"The Computing Landscape of the 21st Century"
Satyanarayanan, M., Gao, W., Lucia, B.
Proceedings of HotMobile 21, February 2019

Adaptation

High-level Choices

When resource supply falls short of demand

1. ***increase supply*** **(resource reservation or QoS)**
2. ***reduce demand*** **(mobile client adaptation)**
3. ***perform corrective action*** **(reconfigure system state)**

Corrective Actions

Application requests system for certain resource level

- **Not satisfiable at the moment**

Systems suggests *corrective action*

- **user/application can perform corrective action**
- **then request resource level again: likely to succeed**
- ***a heuristic, not a guarantee***
- **corrective action may have a cost, and sometimes no reward**
  conceptually similar to a "hint" in distributed systems

Example: Jane's airport scenario from 2001 paper
"Pervasive Computing: Vision and Challenges"

System Role in Adaptation

Odyssey

Application-aware

- system-app partnership
- resource negotiation
- applications modified
- greater complexity

Laissez-faire

Commercial apps

- no system support
- duplicated functionality
- no enforcement of decisions

Application-transparent

Coda

- system fully responsible
- applications unchanged
- perfect for legacy apps

How should applications and the OS partition responsibility for adaptation?

Can reducing fidelity improve user experience?

"Agile Application-Aware Adaptation for Mobility"
Noble, B.D., Satyanarayanan, M., Narayanan, D., Tilton, J.E., Flinn, J., Walker, K.R.
In Proceedings of the 16th ACM Symposium on Operating Systems and Principles, Saint-Malo, France, October, 1997.

ACM SIGMOBILE 2024 Test of Time Award

Fidelity of Data

Adaptation trades *data quality* for *performance* and *resource demand*

Data has a perfect representation: reference copy

- **most current & complete version**
- **what application would see if executed on server**

***Fidelity* = degree to which data matches reference copy**

Fidelity has many dimensions

- **one is universal: consistency**
- **others depend on data type**
  video: frame rate, frame quality; map: feature set, minimum feature size
- **tradeoffs are application-dependent**

How to efficiently support diverse concurrent notions of fidelity?

Agility of Adaptation

Ideal adaptive system

- **perfect, instantaneous knowledge of resource availability**
- **instantaneous reaction by appropriate changes in fidelity**

Agility = speed and accuracy of reaction to resource changes

Complex property with many components

- **system may be much more sensitive to certain resources**
  e.g. bandwidth vs. battery power level
- **resource availability may change for different reasons**
  external (supply) vs.. internal (demand) changes in availability
  different mechanisms may detect these changes
- **determines most turbulent environment acceptable**

Resource Negotiation API

Application specifies resource of interest

- **if successful, applications get a request_id**
- **if resource already out of bounds, current value returned**

Applications give system a *window of tolerance* for resource

- **system monitors resource availability**
- **if it leaves window, notifies application via upcall**

Resource	Units
Network Bandwidth	bytes/second
Network Latency	milliseconds
Disk Cache Space	MB
CPU	SPECints
Battery Life	minutes
Money	cents

Other, type-specific resources also possible

e.g.: Databases that sell subscriptions of N queries per day

Notification of Resource Changes

OS extension monitors resource availability

Generates *upcall* if any resource strays beyond tolerance window

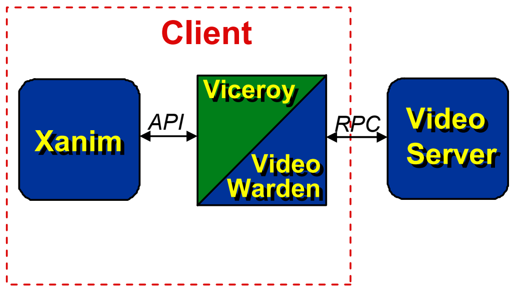
Upcalls similar to signals but

- have exactly-once delivery semantics
- can pass parameters and return results

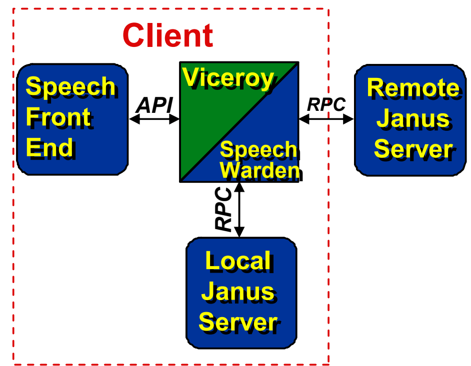
Upcall handers invoked with three parameters

- request to which this notification is the response
- the resource whose level has changed
- the new level of availability

Example Odyssey Applications

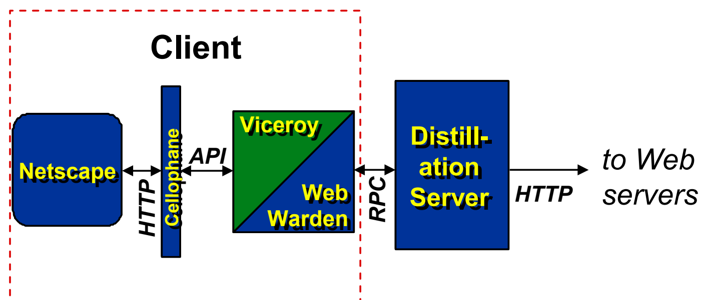


1. Streaming Video



2. Speech Recognition

3. Web Browsing



Does this adaptation stuff from 25 years ago still matter?

Who bothers to do this these days?

Fast Forward to 2022

BumbleBee: Application-aware adaptation for edge-cloud orchestration

HyunJong Lee
University of Michigan

Shadi Noghabi
Microsoft Research

Brian D. Noble
University of Michigan

Matthew Furlong
Microsoft

Landon P. Cox
Microsoft

December 2022
ACM/IEEE Symposium on Edge Computing

***Abstract*—Modern developers rely on container-orchestration frameworks like Kubernetes to deploy and manage hybrid workloads that span the edge and cloud. When network conditions between the edge and cloud change unexpectedly, a workload must *adapt* its internal behavior. Unfortunately, container-orchestration frameworks do not offer an easy way to express, deploy, and manage adaptation strategies. As a result, fine-tuning or modifying a workload's adaptive behavior can require modifying containers built from large, complex codebases that may be maintained by separate development teams. This paper presents BumbleBee, a lightweight extension for container-orchestration frameworks that separates the concerns of application logic and adaptation logic. BumbleBee provides a simple in-network programming abstraction for making decisions about network data using application semantics. Experiments with a BumbleBee prototype show that edge ML-workloads can adapt to network variability and survive disconnections, edge stream-processing workloads can improve benchmark results between 37.8% and 23x, and HLS video-streaming can reduce stalled playback by 77%.**

I. Introduction

Hybrid workloads that span edges and clouds are on the rise [23], [48]. Container technologies like Docker [50] and orchestration platforms like Kubernetes [30] are crucial to hybrid workloads because they provide a uniform compute and

such as a video-processing container that implements adaptive bitrate logic and video transcoding. As a result, fine-tuning or modifying a workload's adaptive behavior can require changes to a large codebase that is often maintained by a separate development team. At the network-transport level application-oblivious responses to variable network conditions, such as TCP congestion control, provide fair bandwidth allocation, but only the application knows how to change its internal behavior as conditions change.

To fill this gap, we present a lightweight in-network processing facility for application-aware adaptation called *BumbleBee*. BumbleBee provides a clean *separation of concerns* between workloads' adaptation and business logic. Workloads' core functionality remain in their original unmodified containers, and BumbleBee adaptation scripts execute in sidecar proxies. BumbleBee benefits a variety of hybrid workloads: ML applications can gracefully switch between high- and low-fidelity inference, stream-processing applications can meet between 37.8% and 23x more deadlines, and video-streaming applications can reduce stalling by 77%.

The main technical challenge that BumbleBee addresses is balancing expressiveness and modularity. Embedding adaptation within an application container enables arbitrary ex-

Shift in focus
Adaptation by cloudlets (Tier-2) rather than by devices (Tier-3)

Predictive Resource Management

Question: Can one do resource-budgeting for applications?

i.e. Give an amount X and say "do the best you can within X"

Leads to creation of *multi-fidelity applications*

- Layered on OS support for predictive resource management

Challenges:

- how do you know what resource consumption will be?
- how data dependent is that consumption?
- how successful are predictions likely to be?

Short answer: past history is a good (not perfect) basis for predictions

Predictive Resource Management for Wearable Computing
Narayanan, D., Satyanarayanan, M.
In Proceedings of MobiSys 2003: The First International Conference on Mobile Systems, Applications, and Services. San Francisco, CA, May, 2003.

Energy Management

Battery life is the slowest-improving resource

next to human intelligence and human attention :-)

Can applications help in extending battery life?

Can application-aware adaptation in Odyssey be applied to energy?

Energy-aware Adapation for Mobile Applications
Flinn, J., Satyanarayanan, M.
In Proceedings of the 17th ACM Symposium on Operating Systems and Principles. Kiawah Island, SC, December, 1999.

ACM SIGMOBILE 2020
Test of Time Award

Viewing battery as a finite but ideal resource (tank) is hard enough

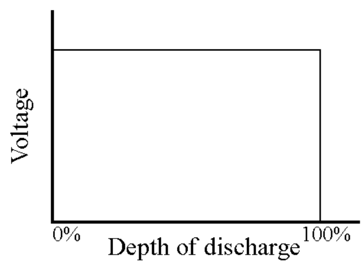
Real batteries are worse!

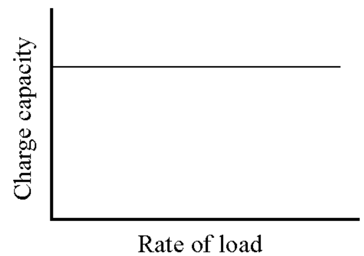
Non-ideal Battery Properties and Low Power Operation
Martin, T.L., Siewiorek, D.P.
In Proceedings of the Third International Symposium on Wearable Computers. San Francisco, CA, October, 1999

Ideal and Real Batteries

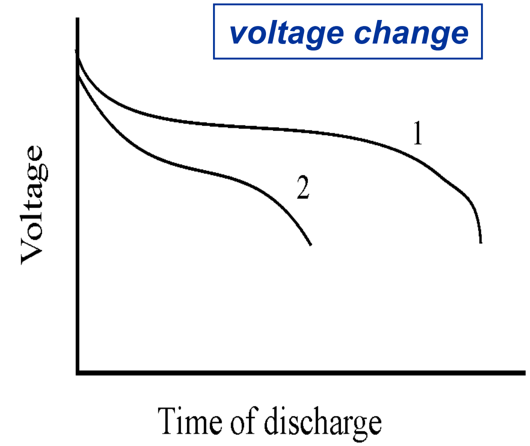
from Martin and Siewiorek 1999

Ideal Battery

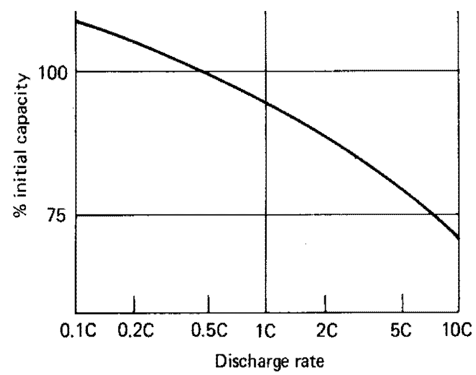




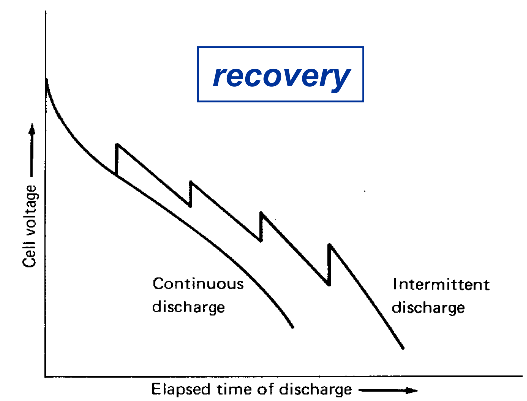
voltage change



loss of capacity



recovery



TakeAway Message

Uncertainty is the bane of mobile computing

This is not going to change anytime soon

Using external resources when possible is one solution (offloading)

Modifying behavior to reduce resource demand is another (fidelity)

Adaptation is key **(mobile entities have to be intelligent, a la Hans Moravec)**