Fundamental Challenges in Mobile Computing

M. Satyanarayanan

School of Computer Science

Carnegie Mellon University

Outline

Why is mobile computing different?

What have we done so far?

What fertile regions await exploration?

Caveats

- This is a personal account
- A report from the front lines, not a survey
- Focus is on mobile information access

Part I: What's Different?

Intrinsic Constraints

Mobile elements are ***resource-poor***

- relative to static elements of same era
- weight, power, size constraints

Intrinsic Constraints

Mobile elements are *resource-poor*

- relative to static elements of same era
- weight, power, size constraints

Mobility hurts *security & robustness*

- theft, destruction more likely
- greater exposure to subversion

Intrinsic Constraints

Mobile elements are *resource-poor*

- relative to static elements of same era
- weight, power, size constraints

Mobility hurts *security & robustness*

- theft, destruction more likely
- greater exposure to subversion

Mobility leads to *communication uncertainty*

- enormous variation in bandwidth & latency
- intermittent connectivity
- may cost real money

Intrinsic Constraints

Mobile elements are *resource-poor*

- relative to static elements of same era
- weight, power, size constraints

Mobility hurts *security & robustness*

- theft, destruction more likely
- greater exposure to subversion

Mobility leads to *communication uncertainty*

- enormous variation in bandwidth & latency
- intermittent connectivity
- may cost real money

Power management is a concern

- actions may have to be slowed or deferred
- communication costs energy

Implications

Contradictory demands on system design

Favoring reliance on servers

- Need for up-to-date information
- Resource-poor clients
- Poorer security & robustness of clients
- Data management at large scale

Favoring ability to operate alone

- you may not be able to contact servers
- communication may be expensive

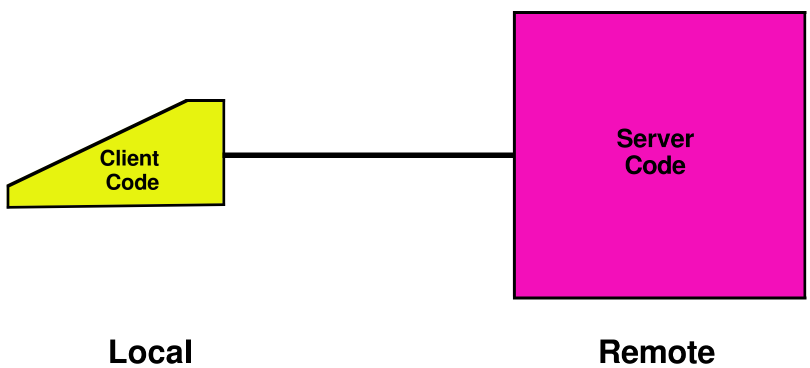
Hence mobile systems must be ***adaptive***

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

Paradigm Shift

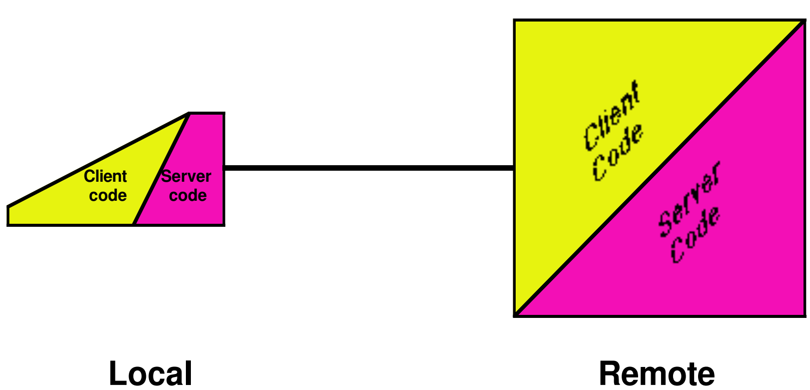
Classic client/server model is inadequate

- static data access model
- function-shipping $\Rightarrow$ dependable connection
- data-shipping $\Rightarrow$ powerful client



Mobility demands extended client/server model

- dynamic choice of function- or data-shipping
- adaptation to load, connectivity, data-type



Part II: Coda and Odyssey

Types of Adaptation

Application-transparent

- **system entirely responsible**
- **applications unchanged**
- ***Coda***

Application-aware

- **system collaborates with applications**
- **resource negotiation**
- **applications modified**
- ***Odyssey***

Key Coda Results

Disconnected operation

- hoarding
- update logging
- reintegration

Optimistic replication

- conflict detection, confinement & repair
- log-based directory resolution
- application-specific file resolution

Support for weak connectivity

- adaptive transport protocols
- rapid cache validation
- trickle reintegration
- model-based cache miss handling

Isolation-only transactions

- detection of read-write conflicts
- resolution mechanisms
- lightweight

Server replication

Coda Status

Deployed since 1991

- ≈ 20 laptops and 15 desktops
- ≈ 25 regular users
- ≈ 4GB of server data
- in regular daily use

Instrumentation to keep track of system usage

- extensive analysis of usage data
- ongoing analysis as new features added

New research results propagated to production system

- first-hand experience with real users
- critical to evaluation of new ideas & mechanisms

Odyssey

Architecture & OS extensions for mobile computing

Components

- **Client-server model for mobile computing**
- **Extensions to Unix API**
- **Internal OS extensions**
- **Sample applications**

API and internal extensions to provide support for

- ***application-aware adaptation***
- ***extensibility***
- ***dynamic sets***

Status

- **early exploration with preliminary prototype**
- **refined implementation in progress**

Part III: Fertile Regions

Disclaimer

Highly speculative

- **no results yet**
- **more questions than answers**
- **tantalizing glimmer of what's possible**

Not exhaustive

- **chosen for possible interest to PODC audience**
- **areas where deep thought & analysis might help**

Caching Metrics

The traditional metric of cache performance is ***miss ratio***

Reliance on miss ratio assumes

- all misses equally important
- all misses require equal service time
- ordering and spacing of misses irrelevant

Mobile computing violates these assumptions

- disconnected user impeded by critical file miss
- miss on big files more expensive
- disk spin down for power saving

Examples of alternative metrics

- references until ***first cache miss***
- references until a ***critical miss***

Some open questions

- What are the right set of metrics?
- Under what circumstances does one use each?
- How does one efficiently monitor these metrics?
- What are the implications for caching algorithms?

Semantic Callbacks

Weak connectivity **makes cache coherence expensive**

- **communication latency**
- **state restoration after intermittent failures**

Coda solution

- **use *callbacks* instead of demand validations**
- **raise *granularity* of cache coherence**
- **trade off *precision of invalidation***
  for *speed of validation*

The notion of callback can be generalized

- **client caches data satisfying predicate P**
- **server remembers *cheap* predicate Q**
- **$Q \Rightarrow P$, but reverse need not be true**
- **Q re-evaluated by server on each update**
- **if Q false, notify client that P might be false**
- **Q is a *semantic callback* for P**

Some open questions

- **When are semantic callbacks most useful?**
- **What forms can P and Q take? Code?**
- **How does one derive Q from P quickly?**
- **Can cost-benefit tradeoff be made adaptive?**
- **Value in speed-of-light limited systems?**

Resource Revocation

Application-awareness makes resource mgmt hard

How to revoke resources granted to application?

Salvation Army Approach	**Give, rather than waste**
United Way Approach	**Give, even if it hurts**
IRS Approach	**Give, or else!**

Some open questions

- **How to handle dirty resources?**
- **What to do when client is disconnected?**
- **What if system not transactional?**
- **How to avoid deadlocks due to revocation?**
- **How to do all this efficiently?**

Analysis of Adaptation

Goal is to maximize ***agility*** while preserving ***stability***

Large body of work on adaptive systems

- subject of ***control theory***
- how applicable are results to mobile systems?

Some open questions

- What are the right metrics of agility?
- How can the agility of a system be improved?
- How does one decide "agile enough"?
- What are the right metrics of system stability?
- What design guidelines ensure stability a priori?

Global Estimation

Mobile client can only make *local observations*

- **e.g., local signal strength**
- **e.g., local packet rate**
- **e.g., average round-trip times**
- **e.g., dispersion in round-trip times**

But adaptation must cope with *distant problems*

- **e.g., overloaded server**
- **e.g., congestion in intermediate network**

No out of band communication

Client must estimate global state from local observations

Some open questions

- **Are there systematic ways to do this estimation?**
- **Can one bound the error?**
- **What is the relationship to agility of adaptation?**
- **Can system support improve global estimation?**

Conclusion

Mobility exacerbates tension between

- *autonomy*
- *interdependence*

Adaptive behavior is key to mobility

- use local resources to reduce communication
- client smarts to cope with uncertainty

Application-transparent adaptation clearly viable *(Coda)*

Jury still out on application-aware adaptation *(Odyssey)*

Mobility is a seminal influence on system design