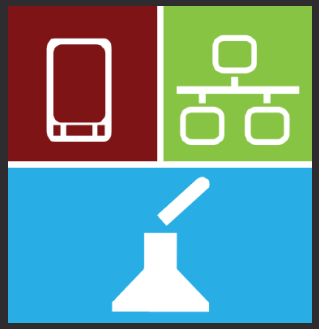


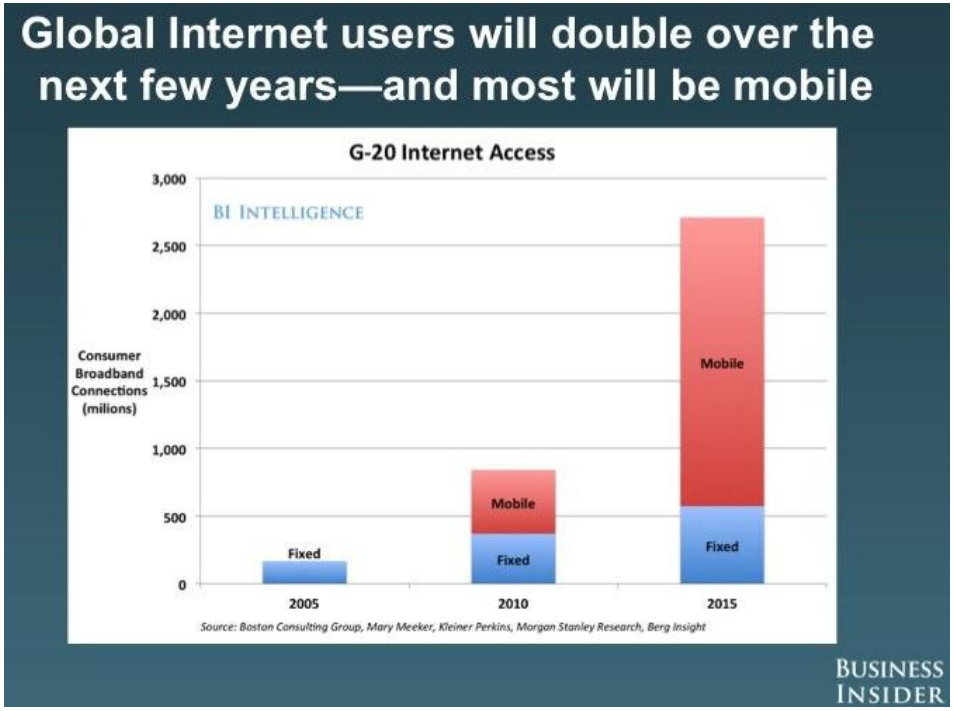
# mobile gaming

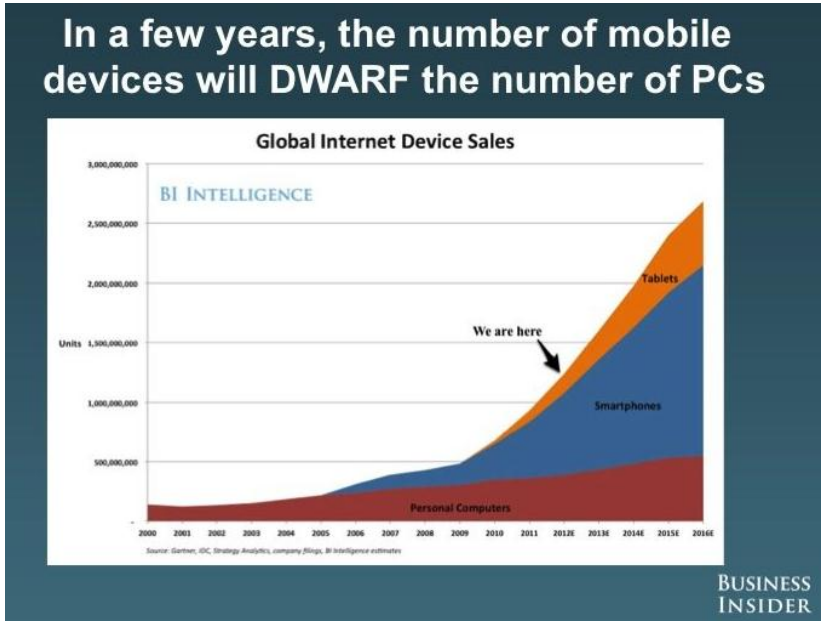
Victor Bahl

8.13.2012

# internet & devices growth (obligatory slide)

Apps are ~$10 Billion market, growing at ~100% per year

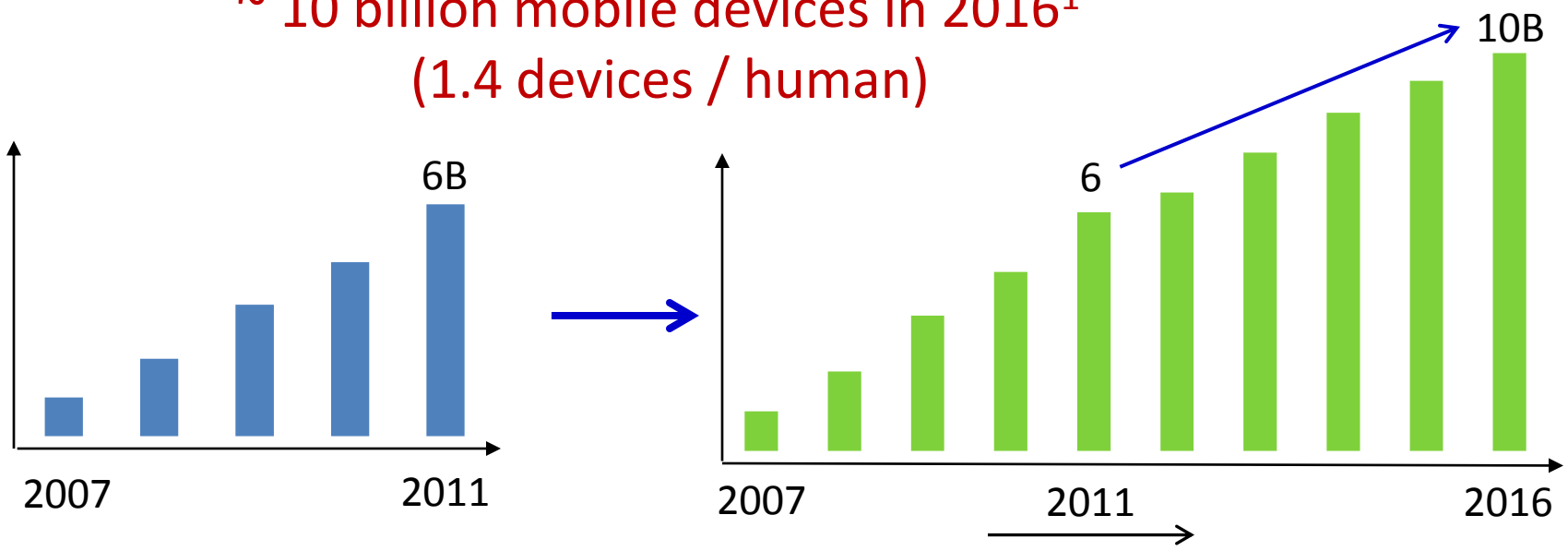



Fun Fact: Getting to 1 M users:
AOL: 9 years; Facebook: 9 months; "Draw Something": 9 days

# bandwidth demand!

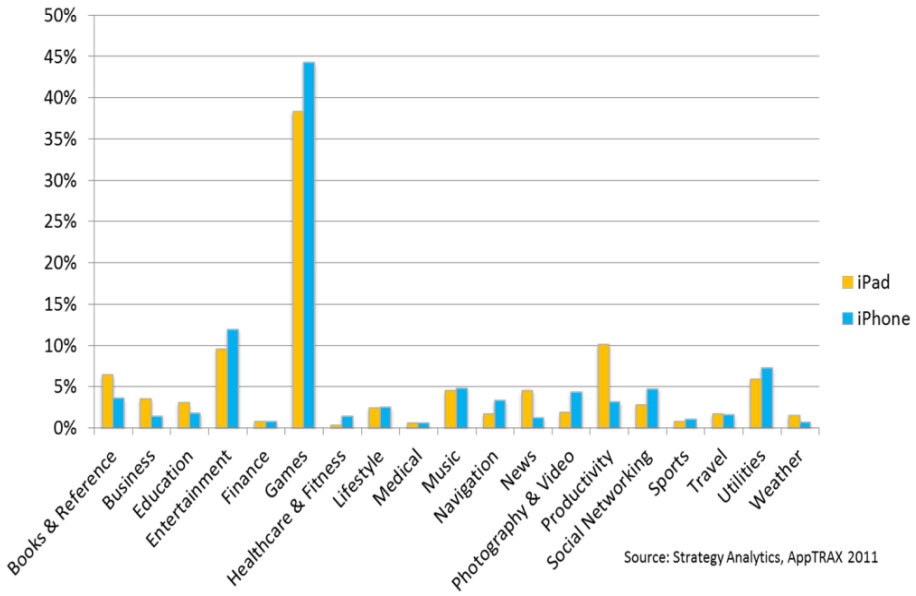
~ 10 billion mobile devices in 2016[1]
(1.4 devices / human)

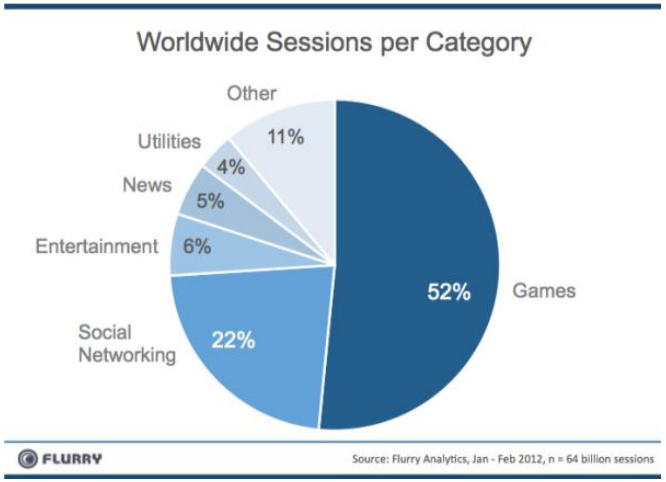



2011-2016 ~ 18X growth in mobile data traffic[2]
(~ 10 exabytes / month)

Source: (1) GSMA; (2) Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011–2016

# gaming today



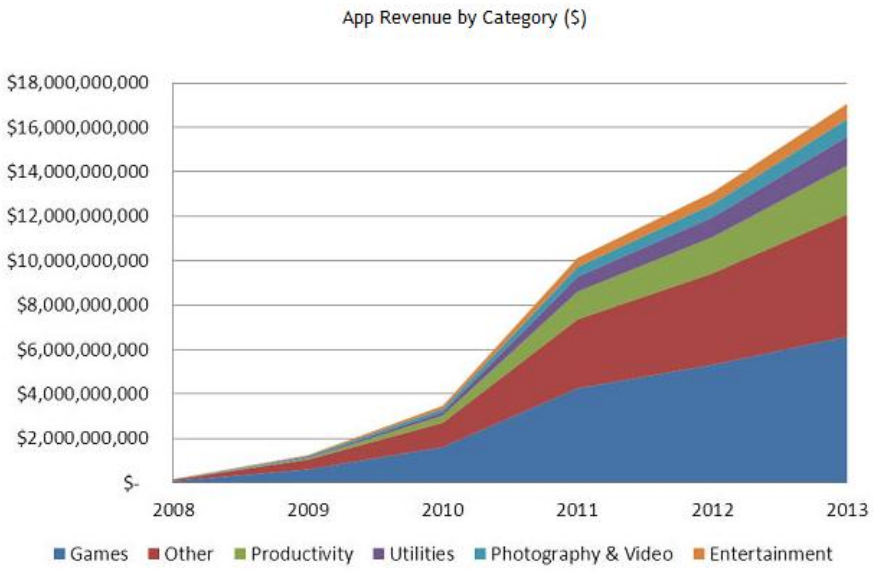

TechCrunch

**Angry Birds Catapults Itself To One Billion Downloads**

INGRID LUNDEN

Wednesday, May 9th, 2012

8 Com




Source: Strategy Analytics - Apptrax

# …but you already knew that ☺

Some things I heard today:

- how game analytics was used to increase dwell time
- how in-the-wild user behavior may be modeled (& used)
- The challenges in getting to MMOG games
- wireless peer-to-peer games
- power management by making use of saliency

All great stuff, let me say a few words about some things I didn't hear …..

# services behind the games

Fun fact: in 2011 ~$12 billion was spent on social/mobile games
in 2015 revenue is projected to be ~24 billion (19% CGR)


Apps that connect to backends receive higher rankings and more downloads because they are likely dynamic with more fresh content and are more social and contextual

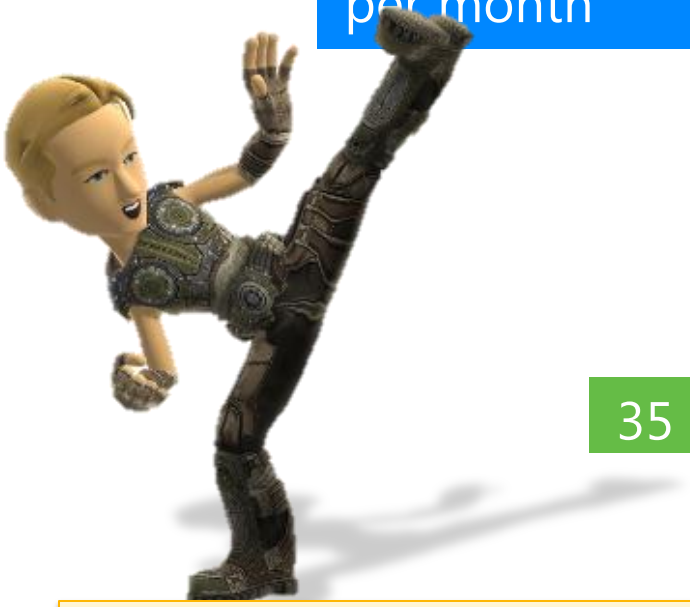
- Kinvey Inc., 2012

# Xbox LIVE

30% growth year over year

40+ Million Users

2.1 billion hours played per month



XBOX LIVE

35 Countries

176,802,201,383 Gamer Points scored

In US: subscribers spend 84 hours/month

Since 2010, $$ spent advertising has increased 142%

# Xbox LIVE is coming to Windows 8

Xbox LIVE provides a comprehensive set of services for Windows 8 game developers that are proven to be useful

https://services.xboxlive.com

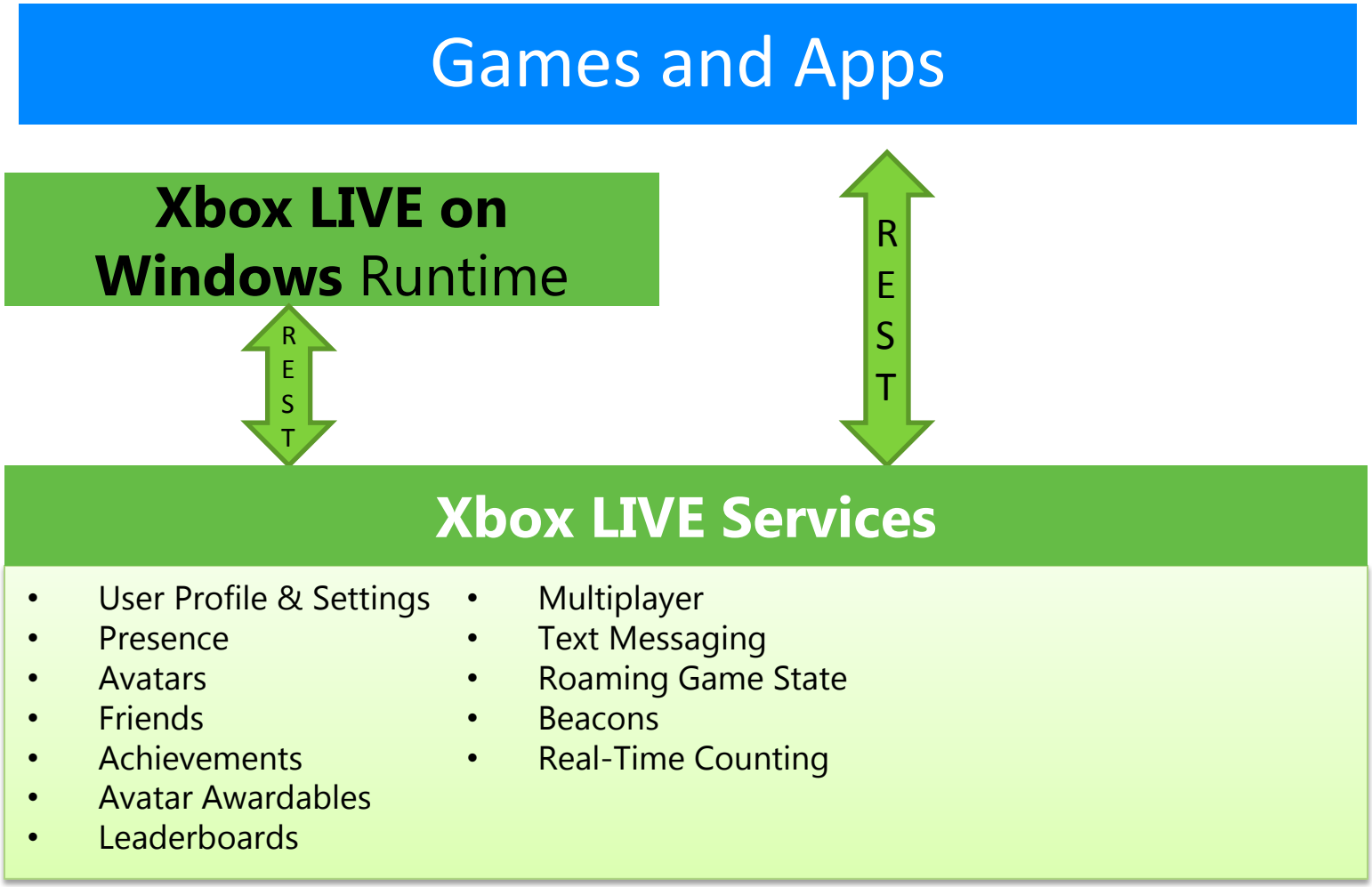
# XBox SmartGlass

## services to enable multi-device gaming experience

HTML5 apps are pushed to your device via a backend service

# Xbox LIVE services

## first generation of cloud services for games



# let’s talk about -
# next gen. services

# perennial problems

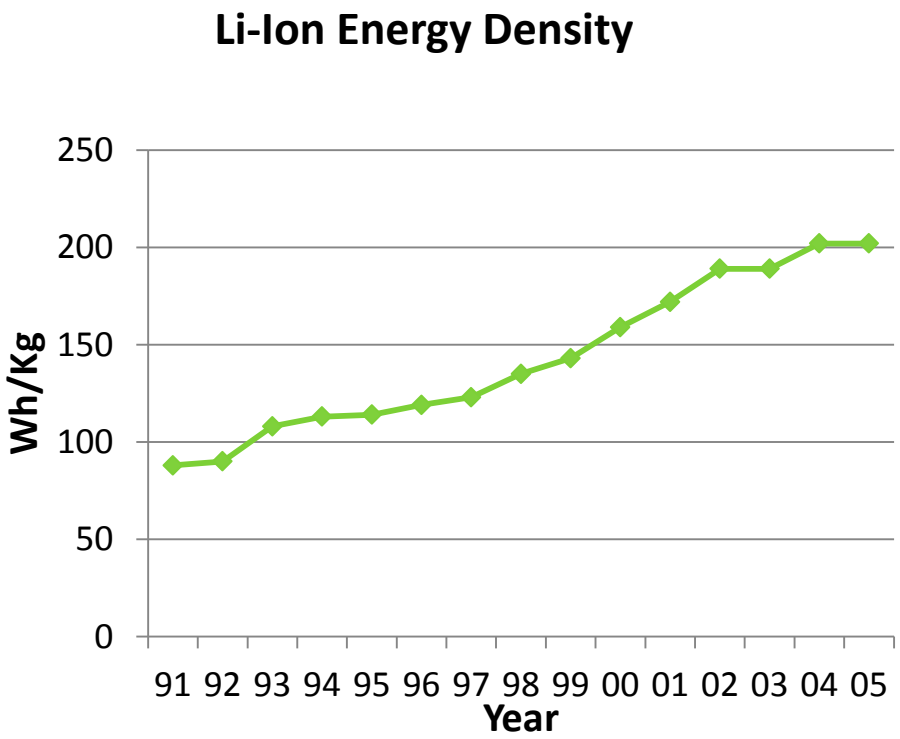
- network bandwidth
- battery life
- computation latency
- network latency
- fine-grain localization
- sensor accuracy, ….

Take a look at MobiGames'12 Chairs' welcome message

Several panelist spoke about these as well

compute and energy limitations can destroy mobile gaming experience

# energy scarcity: silver bullet seems unlikely




## lagged behind

- Higher voltage batteries (4.35 V vs. 4.2V) – 8% improvement
- Silicon anode adoption (vs. graphite) – 30% improvement

## trade-offs

- Fast charging = lower capacity
- Slow charging = higher capacity

Contrast with

CPU performance improvement during same period: 246x

# compute scarcity

Augmented Reality

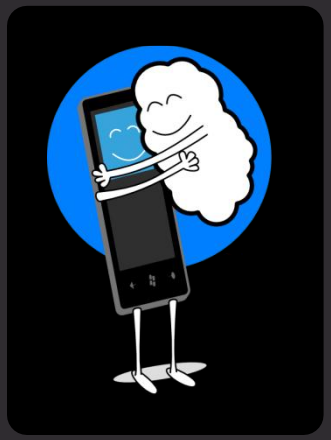
Too CPU intensive


3D Interactive Gaming

Not on par with desktop counterparts


## ….limits the gamer's experience

# promising direction: offload computation

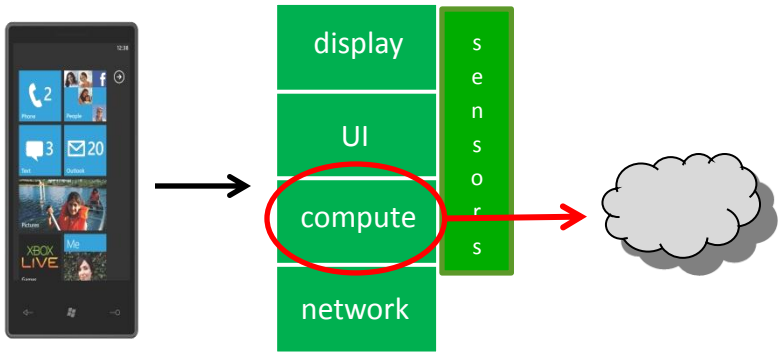


remote execution can reduce energy consumption and improve performance

# opportunistic use of the cloud

## research challenges

- what to offload?
- how to dynamically decide when to offload?
- how to minimize programmer effort?




## important for adoption: a simple programming model

- app developer community has varying expertise & skills
  - Cannot require app developers to become experts in distributed systems

"I just want to write game logic on the server – I don't want to be concerned with scaling, DBs, figuring out how many servers I need, etc.

-- Game Developer-Magazine (Survey of Mobile & Social Technology, May 2012 Issue)

# programming model choices

- MAUI: exploits .NET framework to dynamically partitioning & offload method execution [Microsoft, MobiSys'10]
- Odessa: creates a data-flow graph to exploit parallelism [USC, MobiSys 2011]
- CloneCloud: supports existing applications, but requires tight synchronization between cloud and phone [Intel, EuroSys 2011]
- Orleans: a new programming model based on grains [Socc'11]

| | MAUI | CloneCloud | Odessa | Orleans |
|---|---|---|---|---|
| Remote execution unit | Methods (RMI) | Threads | Tasks | Grains |

# enabling simple program partitioning

Programming Model

- Dynamic partitioning made simple for th partitioning
  - Programmer builds app as standalone phone app
  - Programmer adds .NET attributes to indi "remoteable" methods / classes

```
[Remoteable]
ArrayList GetValidMoves(Square s)
{
    if (s.IsEmpty())
    {
        return new ArrayList();
    }
    if (s.Piece.IsEnemyOf(active))
    {
        //this piece does not belong to the active side, no moves possible
        return new ArrayList();
    }
    //forward the call to the Rule-class
    return rules.getMoves(s);
}
```

Salient Point:
The model supports disconnected operations

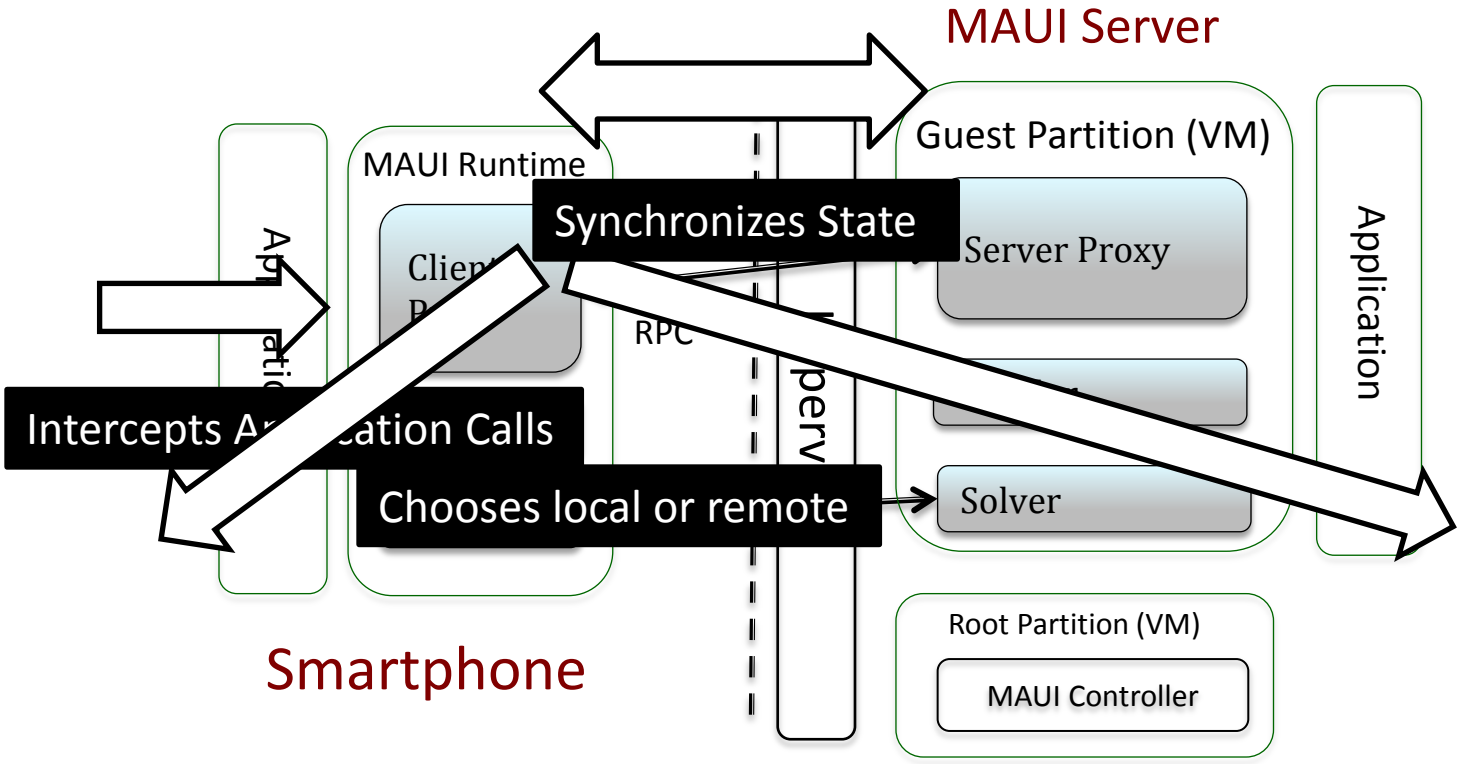
- MAUI runtime: partitions (splits) the program at run-time
  - Can optimize for energy-savings, or performance

Why not use a static client/server split?

- Developers need to revisit application structure as devices change
- Failure model: when phone is disconnected, or even intermittently connected, applications don't work
- The portion of an app that makes sense to offload changes based on the network conn. to the cloud server

# dynamic offloading

## Application Partitioning

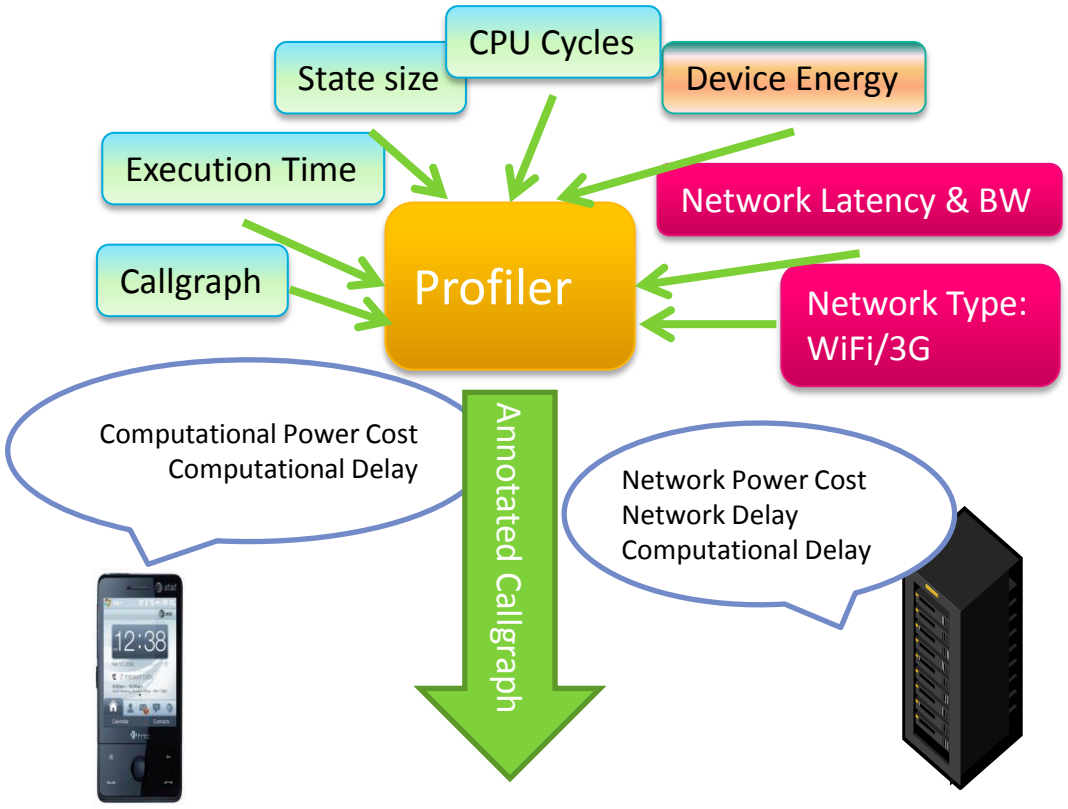



client/server split, can be extended to multiple tiers

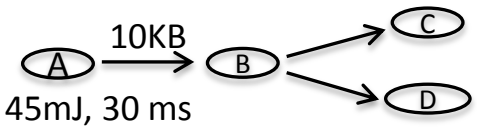
# profiler and decision engine

## Profiler:

Handles dynamics of devices, program behavior, and environment (Network, Server Load)




- Vertex: method annotated with computation energy and delay for execution
- Edge: method invocation annotated with total state transferred




## Decision Engine:

Partition A Running App

We use an Integer Linear Program (ILP) to optimize for performance, energy, or other metrics…

Example – Maximize:

$$\sum_{v \epsilon V} ( I_v \times E_v ) \quad - \quad \sum_{(u,v) \epsilon E} ( |I_u - I_v| \times C_{u,v})$$

energy saved — cost of offload

Such that:

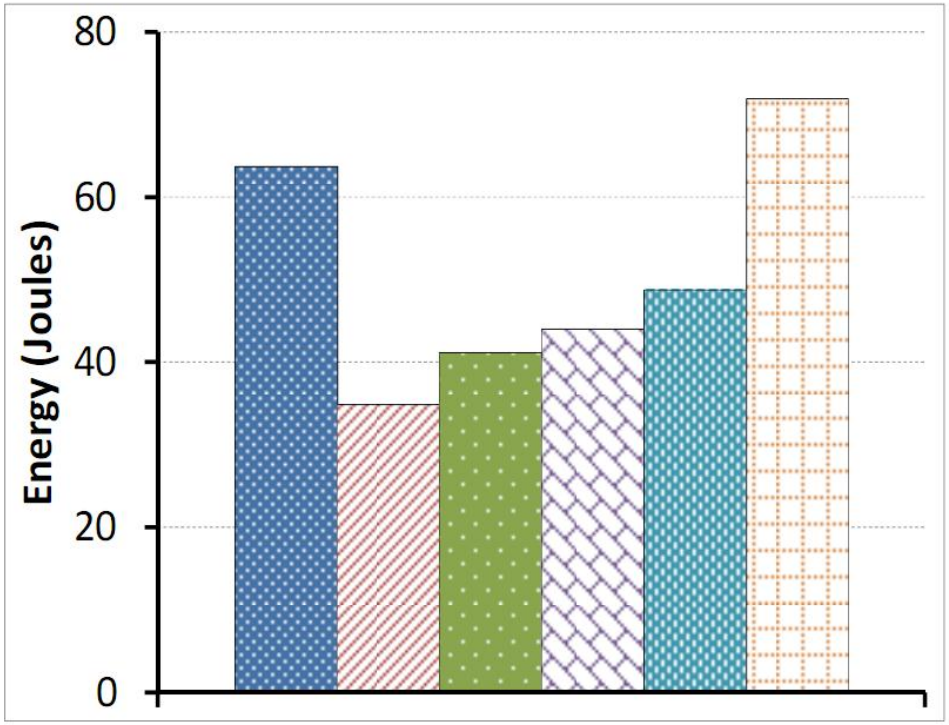
$$\sum_{v \epsilon V} ( I_v \times T_v ) \quad + \quad \sum_{(u,v) \epsilon E} ( |I_u - I_v| \times B_{u,v}) <= \text{Lat.}$$

execution time — time to offload

and

$I_v <= R_v$ for all $v \epsilon V$

# performance benefits




Energy Benefits:

Interactive arcade game w/physics engine:

Energy measurements from hardware power monitor


Arcade game benefits:

- Up to double the frame rate
- Up to 40% energy reduction

# How about a service that virtualizes the screen?

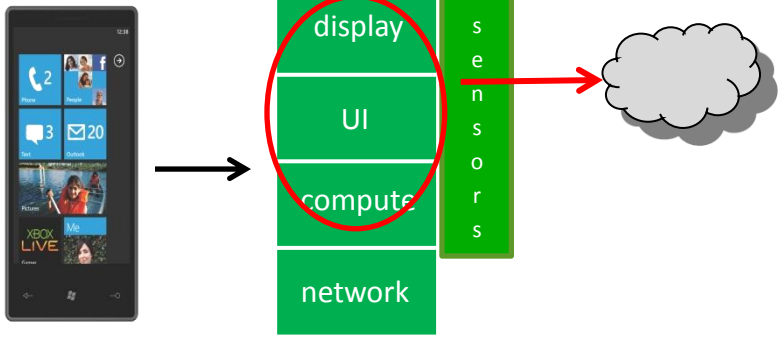
DataCenters

A real-time, low-delay cloud server technology
for remoting display rendering and user interactions

# offloading in the real-world

existing players:






offload the CPU- and GPU-intensive tasks to a remote render farm, then beam the gameplay as a streaming video

*Comments*

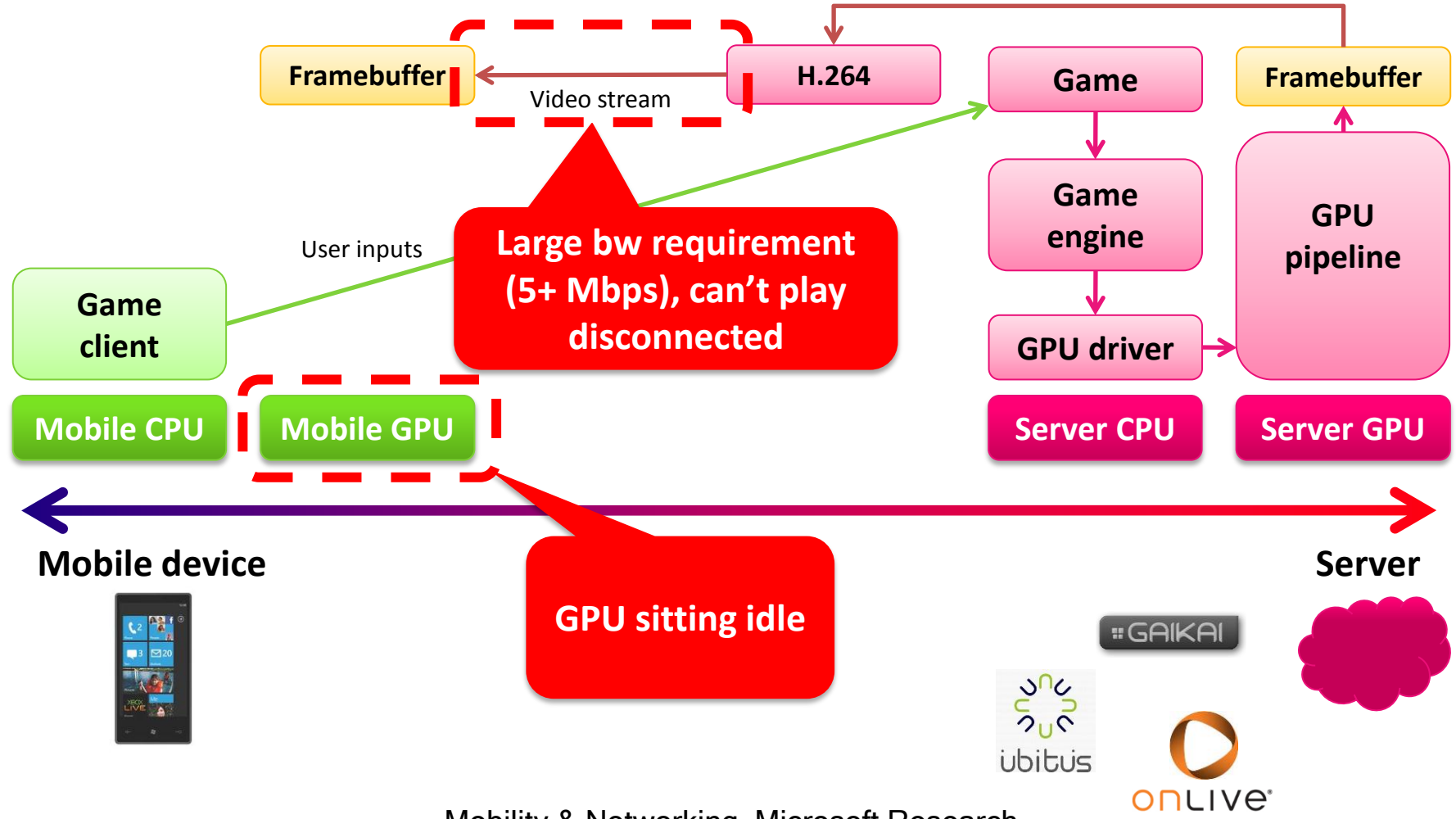
c|net

*"System requires a 3-5Mbps connection to work, and can be unforgiving of spotty Internet access speeds. Wi-Fi is still wonky on the PC client"*

PC PCMAG.COM

*"If your machine is connected via Wi-Fi, an error message will pop up. System requirements include a dual-core microprocessor and a 5-Mbit/s wired connection."*

# thin-client approach



can we use the mobile GPU to save bandwidth?

# collaborative rendering

basic approach

- client computes **low-fidelity** output using mobile GPU
- server ships additional information
- client combines information sources for **high-fidelity** result

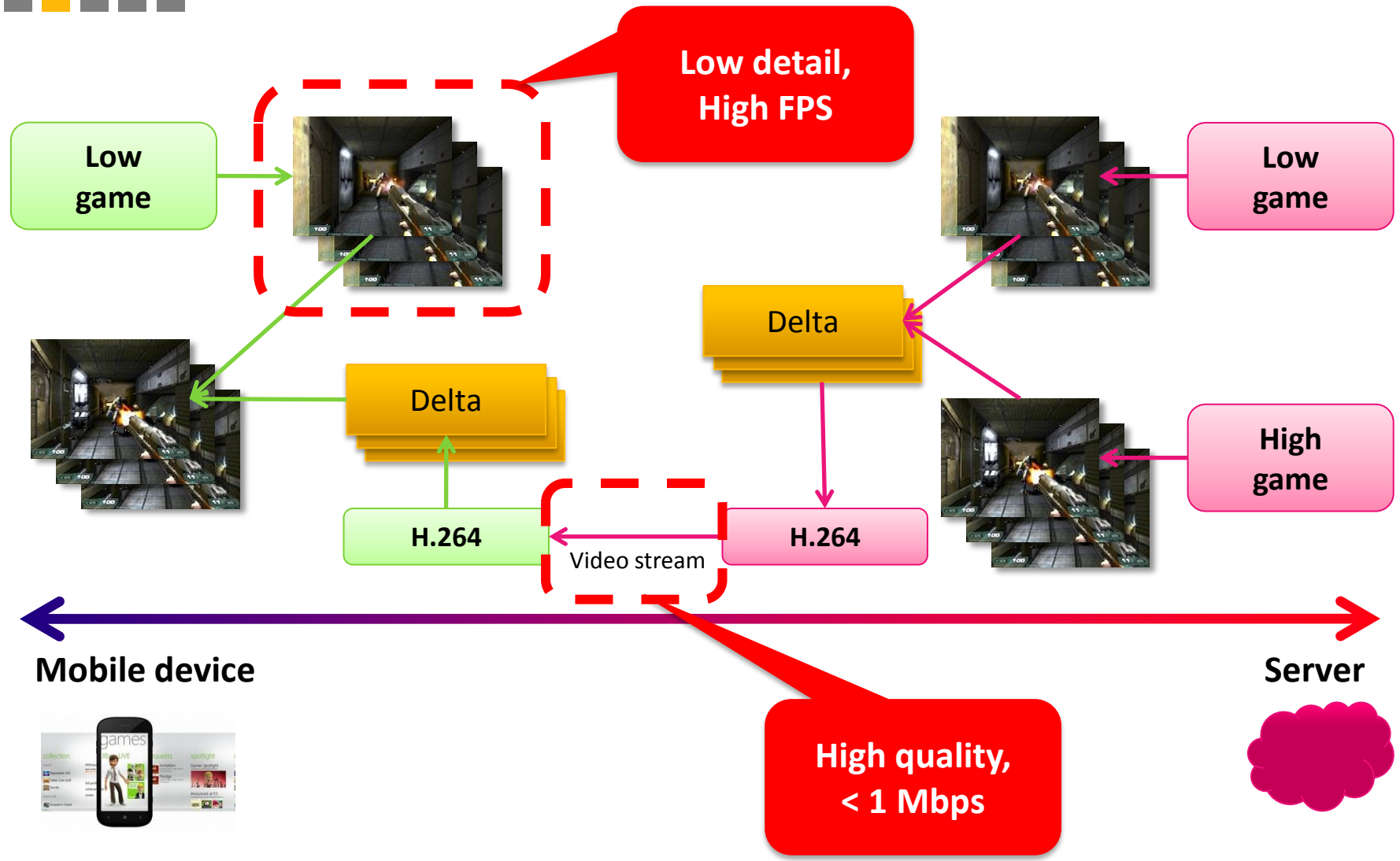
intuition of why it would work

- low fidelity game output contains **most** scene information
- missing details are expensive to compute, but missing details are relatively small percent of total info

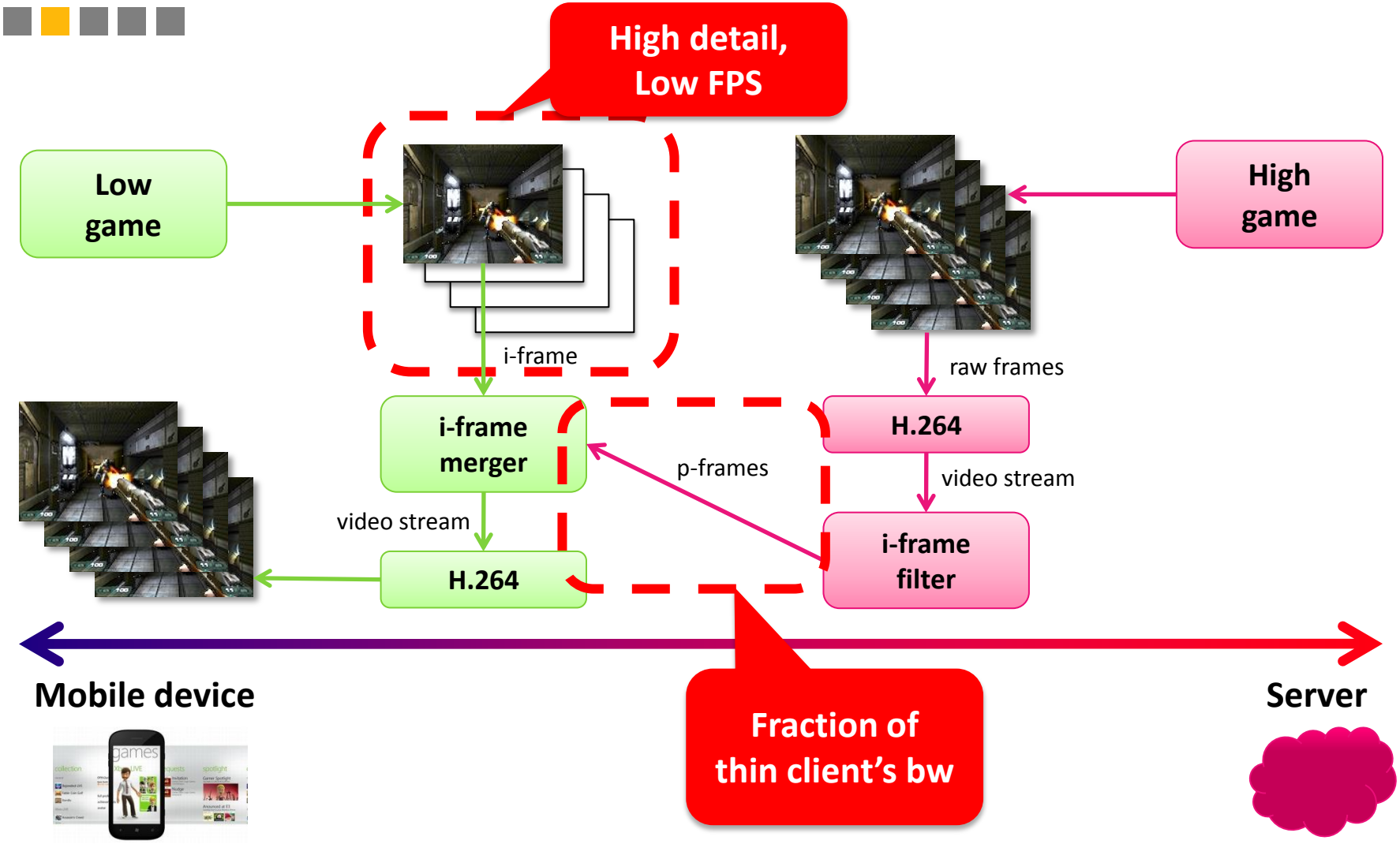
# Doom 3 (low fidelity)

# Doom 3 (high fidelity)

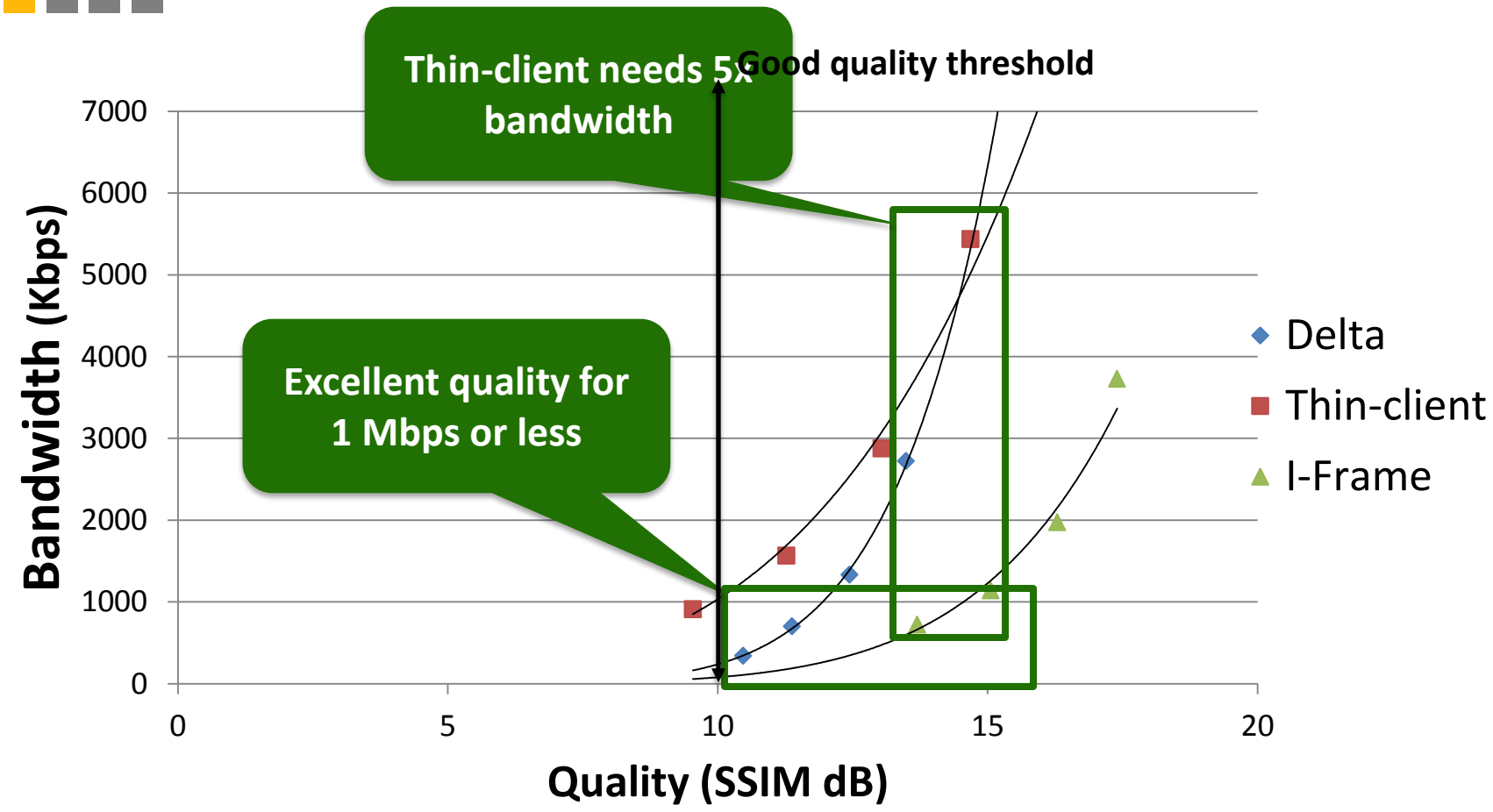
# Approach 1: delta encoding



# Approach 2: i-frame rendering



# bandwidth versus quality




SSIM: Structural Similarity Index Metric

# summarizing

## offloading should be cloud service

- code offload allows developers to bypass resource limitations of handheld devices
- with dynamic offload, programmers no longer worry about *where* their code runs
- Encourages developers to build applications they would never have considered possible
- ..but for all this to work we need to be able to scale

"There seriously needs to be an entirely new category of commodity internet infrastructure designed for scaling games "

-- Game Developer-Magazine (Survey of Mobile & Social Technology, May 2012 Issue)

generic offloading should be cloud service, but what else?

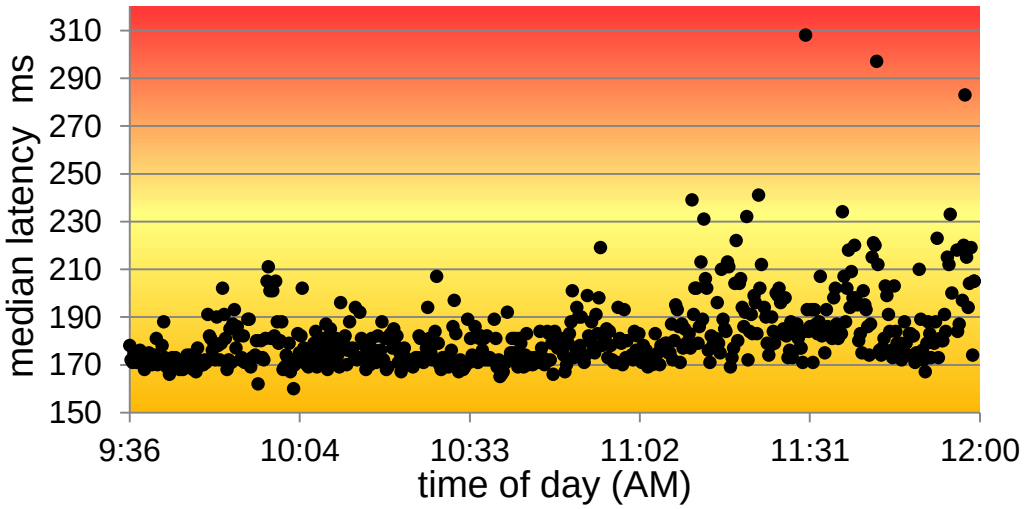
# how about a service that helps match up gamers

| type | latency threshold |
|---|---|
| Halo3 | ≈ 150 ms |
| sports | ≈ 500 ms |
| strategy | ≈ 1000 ms |

| location | median latency |
|---|---|
| Los Angeles | ≈ 120 ms |
| Durham | ≈ 175 ms |
| Kauai | ≈ 180 ms |
| Seattle | ≈ 180 ms |

# the match-making cloud service

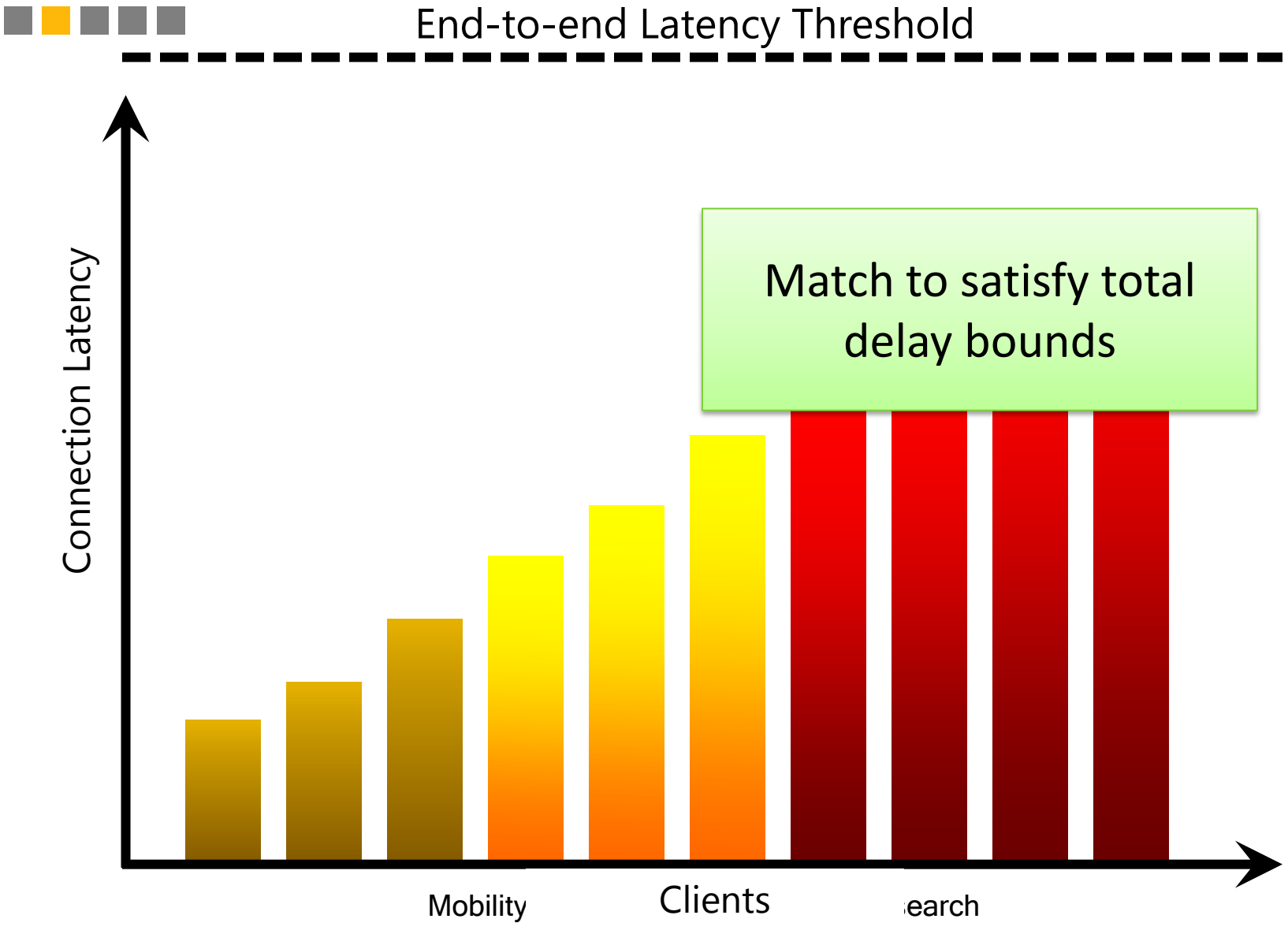
- matchmaking abstracts away 2 hard problems for game developer
  1. estimating network latency of players
  2. grouping of players into viable games




3G measurement study:

- Phone-to-phone latency stable over 15 minute intervals
- Can share latency profiles between phones using same cell tower

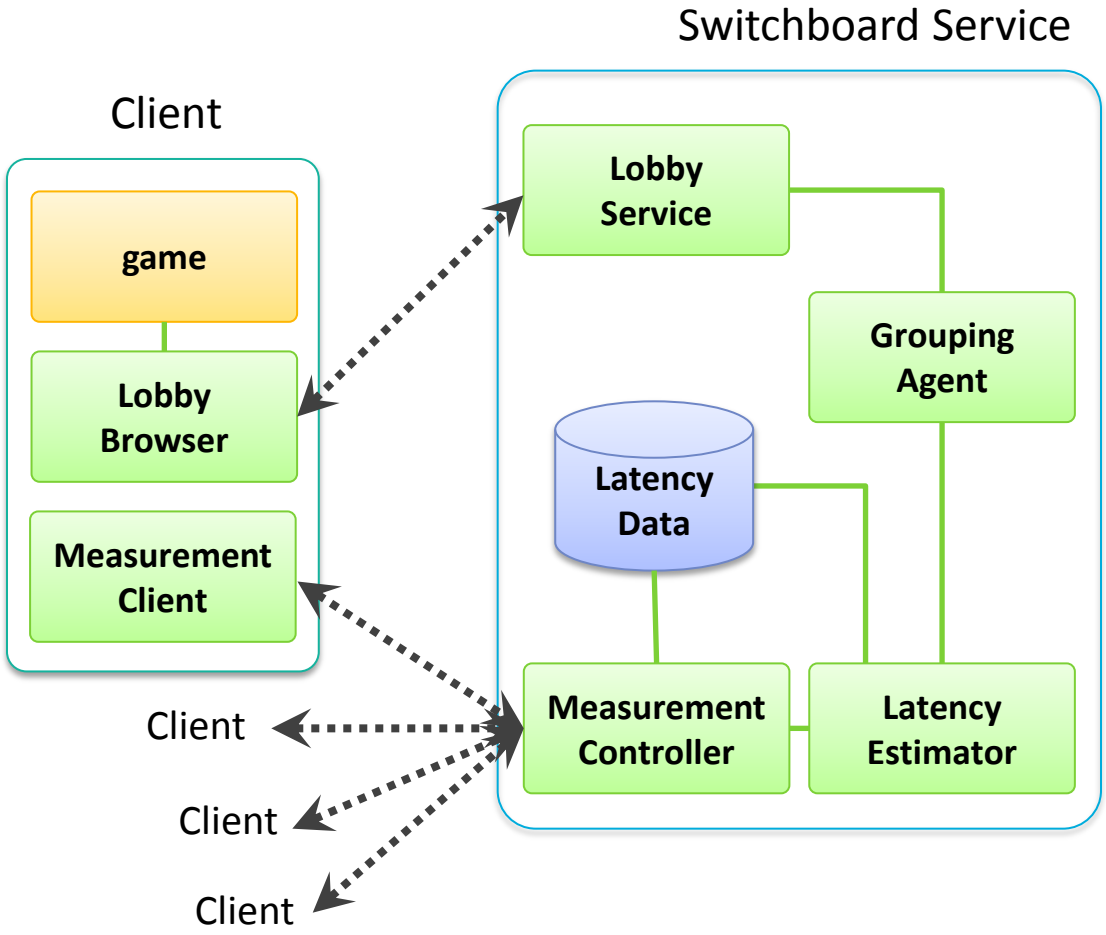
# the matchmaking problem



# switchboard: matchmaker service

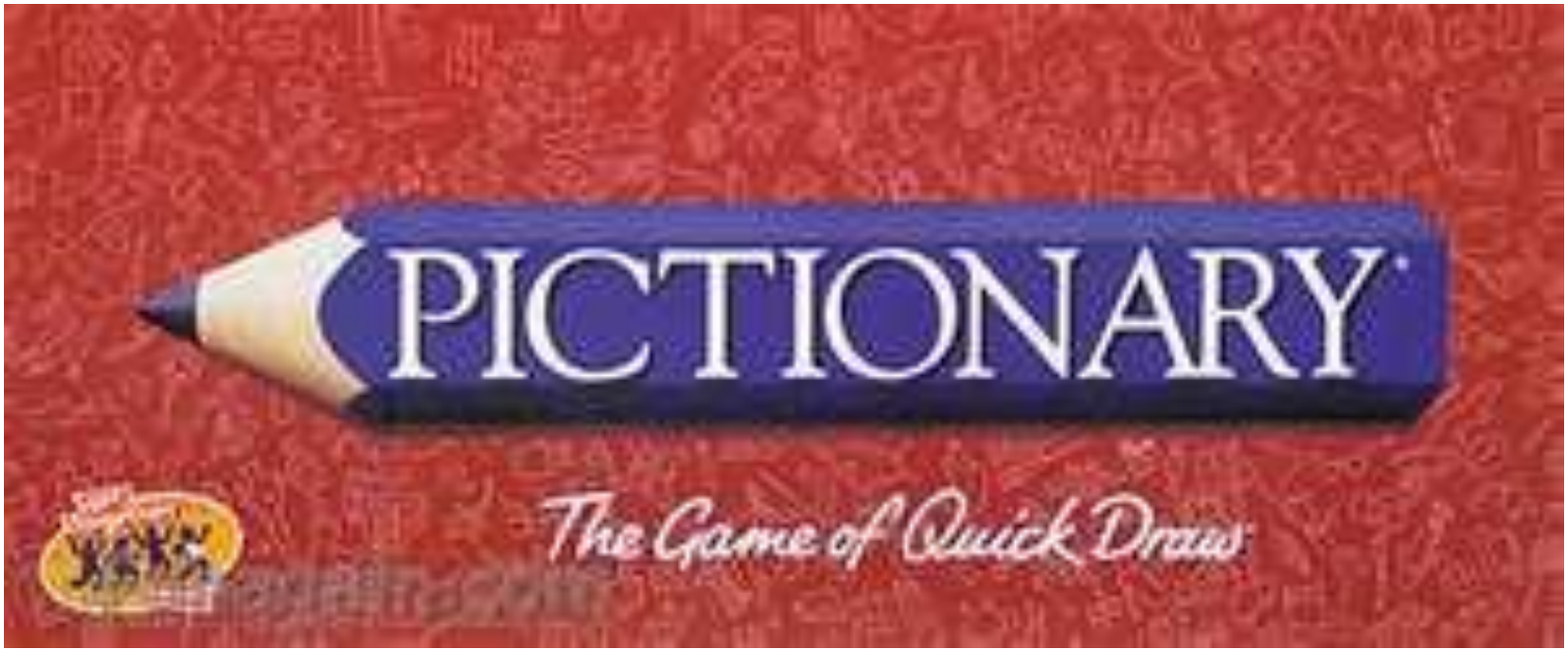
matchmaking for mobile devices is much harder

- cellular latency is highly variable
- scale is larger (e.g. # of phones vs. # of consoles)



# Example game (old)

# Example: relay service



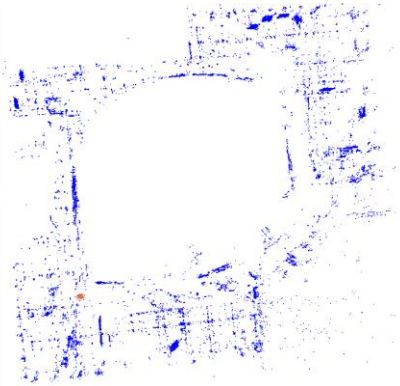
# future: phones that see

Looxcie, Inc


who?



what?



where?

Video credits:
Matthai Philipose, MCRC
Intel Labs

# “service store” for game developers

… build world-class cloud services that enable game application developers to easily realize the full potential of their vision

Examples:

- computation offload
- rendezvous: Lookup for relay endpoints
- relay: Phone to phone data transfer
- matchmaking
- social mobile sharing for ad hoc groups
- gesture recognition
- object recognition
- ….

Toolbox of services

sophisticated resource intensive algorithms running in the cloud typically CPU, memory & storage intensive battery and/or bandwidth hungry

# latency:
# poor latency can be kill certain mobile games

# thresholds

| Game Type | Latency Threshold |
|---|---|
| First-person, Racing | ≈ 100 ms |
| Sports, Role-playing | ≈ 500 ms |
| Real-time Strategy | ≈ 1000 ms |

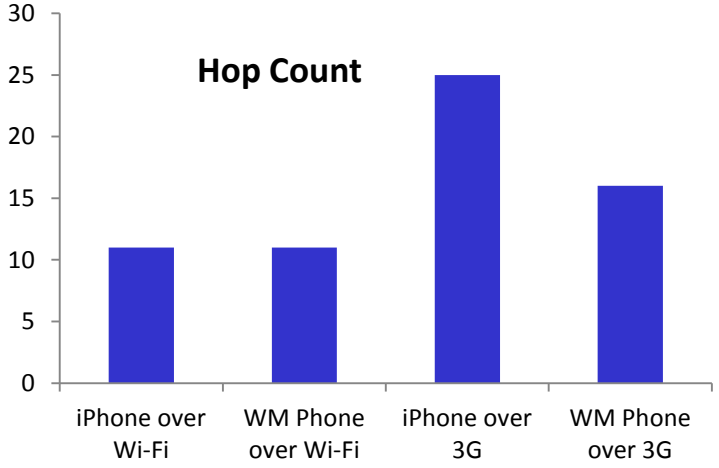
e.g. Halo is a fast action
Low latency game

# latency: a simple experiment

### iPhone via Wi-Fi : 11 hop

Wi-Fi -> 209.85.225.99

1. (10.0.2.1) 8.513 ms 8.223 ms 9.365 ms
2. (141.212.111.1) 0.913 ms 0.606 ms 0.399 ms
3. (192.122.183.41) 11.381 ms 6.054 ms 5.975 ms
4. (192.12.80.69) 7.038 ms 7.353 ms 7.026 ms
5. (198.108.23.12) 12.525 ms 13.027 ms 12.619 ms
6. (198.110.131.78) 12.715 ms 9.424 ms 9.315 ms
7. (216.239.48.154) 9.974 ms (209.85.250.237) 10.295 ms (216.239.48.154) 9.405 ms
8. (72.14.232.141) 19.308 ms 22.249 ms 23.312 ms
9. (209.85.241.35) 32.987 ms 22.708 ms (209.85.241.27) 124.588 ms
10. (72.14.239.18) 22.256 ms (209.85.248.106) 29.154 ms (209.85.248.102) 21.635 ms
11. (209.85.225.99) 19.973 ms 21.930 ms 21.656 ms



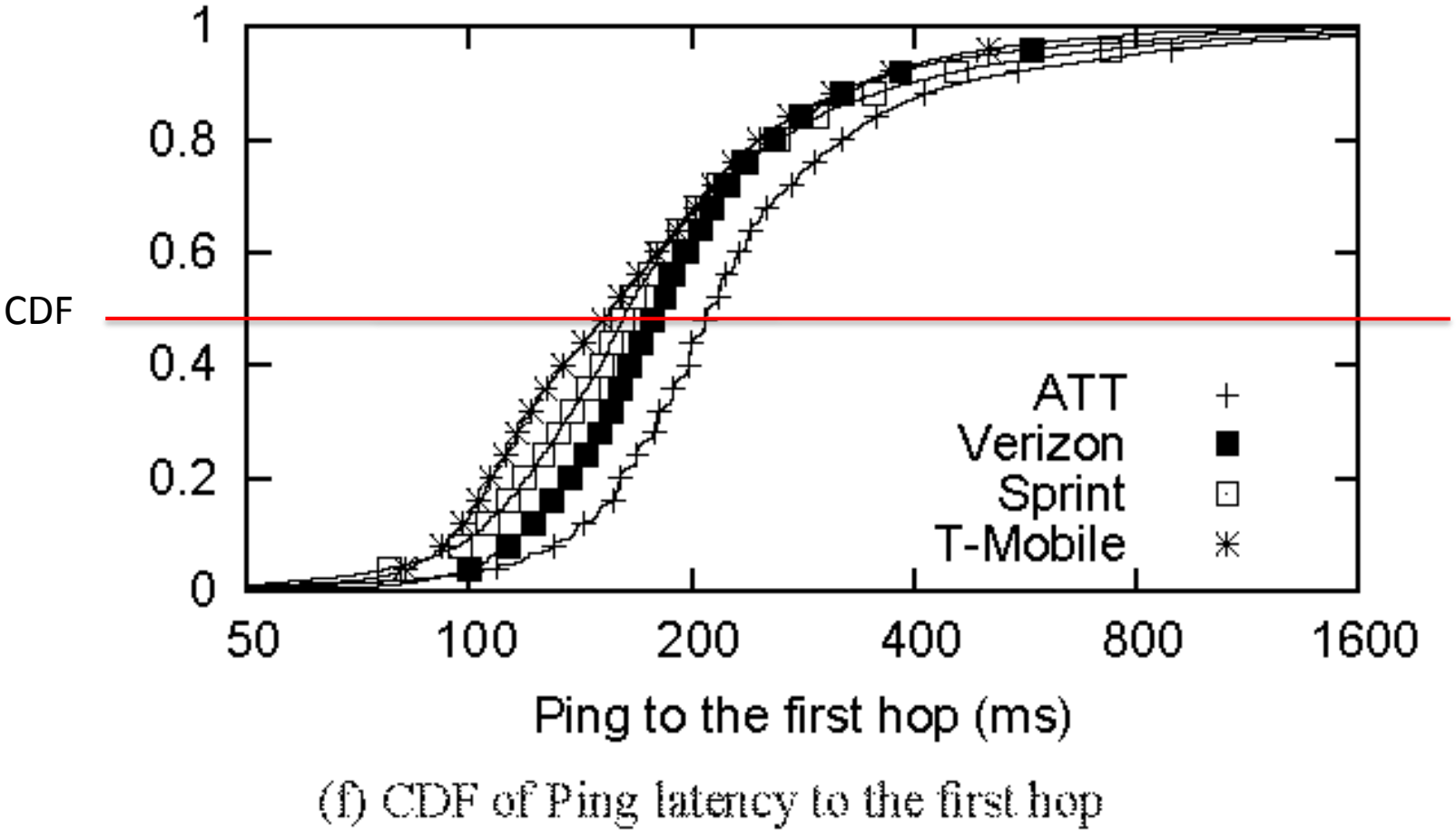

### iPhone via 3G : 25 hop

3G -> 209.85.225.99

1. * * *
2. (172.26.248.2) 414.197 ms 698.485 ms 539.776 ms
3. (172.16.7.82) 1029.853 ms 719.595 ms 509.750 ms
4. (10.251.11.23) 689.837 ms 669.340 ms 689.739 ms
5. (10.251.10.2) 509.781 ms 729.746 ms 679.787 ms
6. (10.252.1.7) 719.652 ms 760.612 ms 788.914 ms
7. (209.183.48.2) 689.834 ms 599.675 ms 559.694 ms
8. (172.16.0.66) 539.712 ms 809.954 ms 689.547 ms
9. (12.88.242.189) 589.857 ms 1129.848 ms 709.784 ms
10. (12.122.138.38) 589.699 ms 1009.723 ms 769.808 ms
11. (12.122.138.21) 669.690 ms 529.758 ms 699.965 ms
12. (192.205.35.222) 699.569 ms 979.769 ms 1489.869 ms
13. (4.68.19.190) 699.435 ms (4.68.19.126) 559.875 ms (4.68.19.62) 499.598
14. (4.69.136.149) 889.946 ms (4.69.136.141) 879.443 ms (4.69.136.145) 469.601 ms
15. (4.69.132.105) 559.716 ms 539.754 ms 1219.982 ms
16. (4.69.132.38) 719.700 ms 659.613 ms 539.695 ms
17. (4.69.132.62) 549.752 ms 549.640 ms 800.128 ms
18. (4.69.132.114) 669.729 ms (4.69.140.189) 769.711 ms 959.663 ms
19. (4.69.140.193) 959.735 ms 979.674 ms 849.886 ms
20. (4.68.101.34) 649.609 ms 659.767 ms (4.68.101.98) 1119.996 ms
21. (4.79.208.18) 669.405 ms 629.574 ms (209.85.240.158) 1200.039 ms
22. (209.85.240.158) 769.538 ms (72.14.232.141) 729.505 ms (209.85.241.22) 719.715 ms
23. (209.85.241.22) 769.665 ms (209.85.241.35) 769.880 ms 859.536 ms
24. (209.85.241.29) 589.710 ms (66.249.95.138) 789.762 ms (209.85.248.106) 913.287 ms
25. (209.85.225.99) 716.000 ms (66.249.95.138) 1039.963 ms (72.14.239.18) 899.607 ms

traceroute to 209.85.225.99 (one of the server IPs of www.google.com)

# RTT comparison for 3G networks

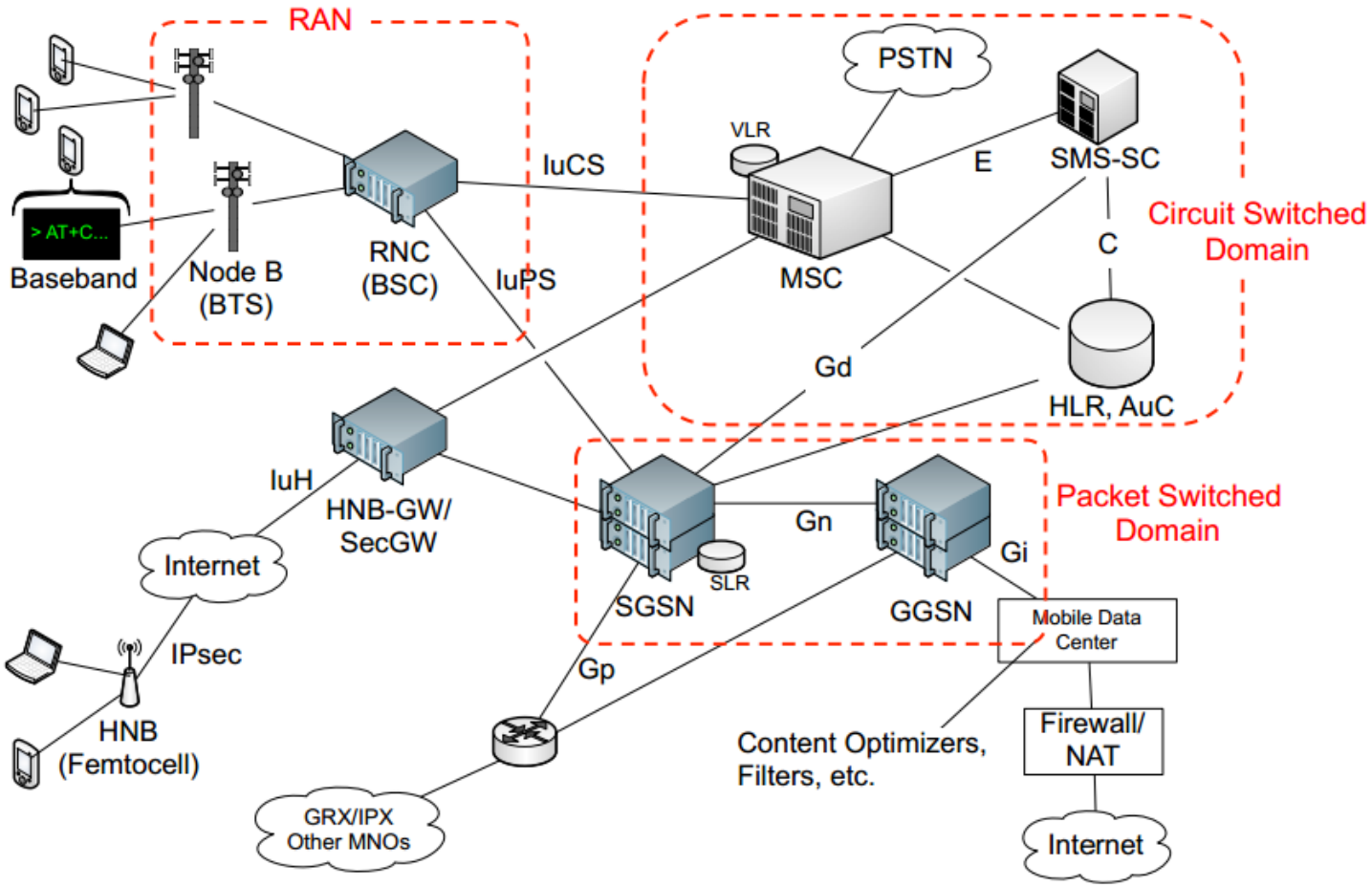



(f) CDF of Ping latency to the first hop

# try it out for yourself: TestMyNet


Available on Windows Phone Maretplace
101 Reviews, average review rating of 4.75/5 stars

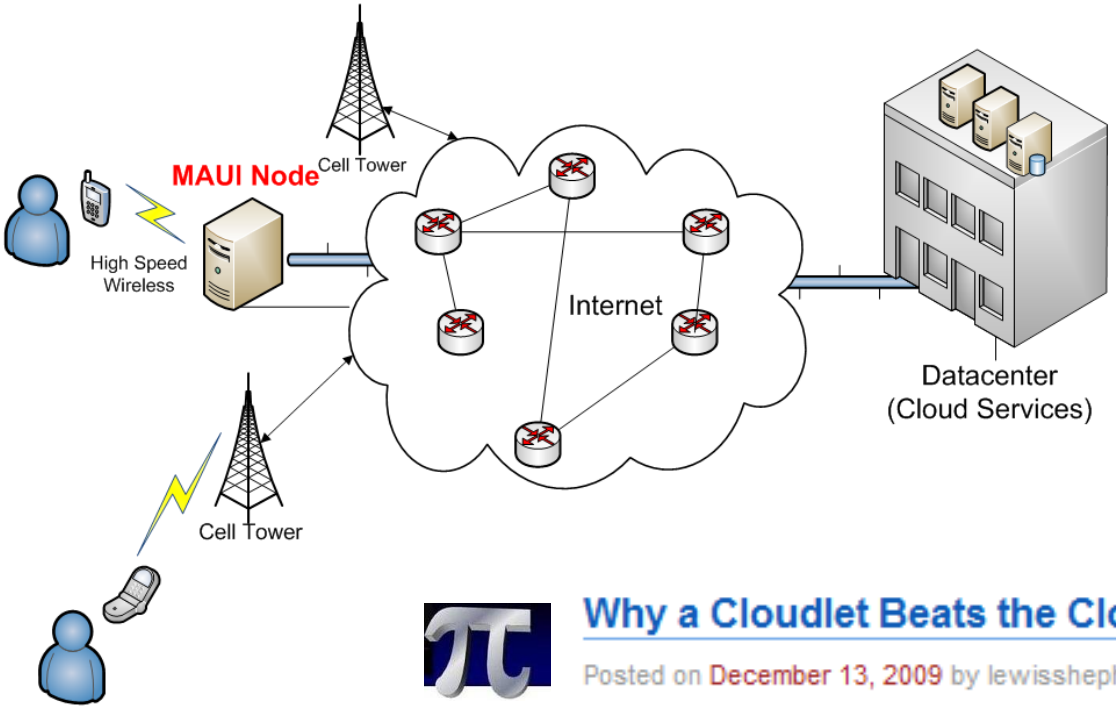
# why? heavyweight architecture



# potential solution: Cloudlets

# reducing latency: cloudlets

a resource rich infra-structure computing device with high-speed Internet connectivity to the cloud that a mobile device can use to augment its capabilities and enable applications that were previously not possible




**Why a Cloudlet Beats the Cloud for Mobile Apps**

Posted on December 13, 2009 by lewisshepherd

# sample deployment scenario

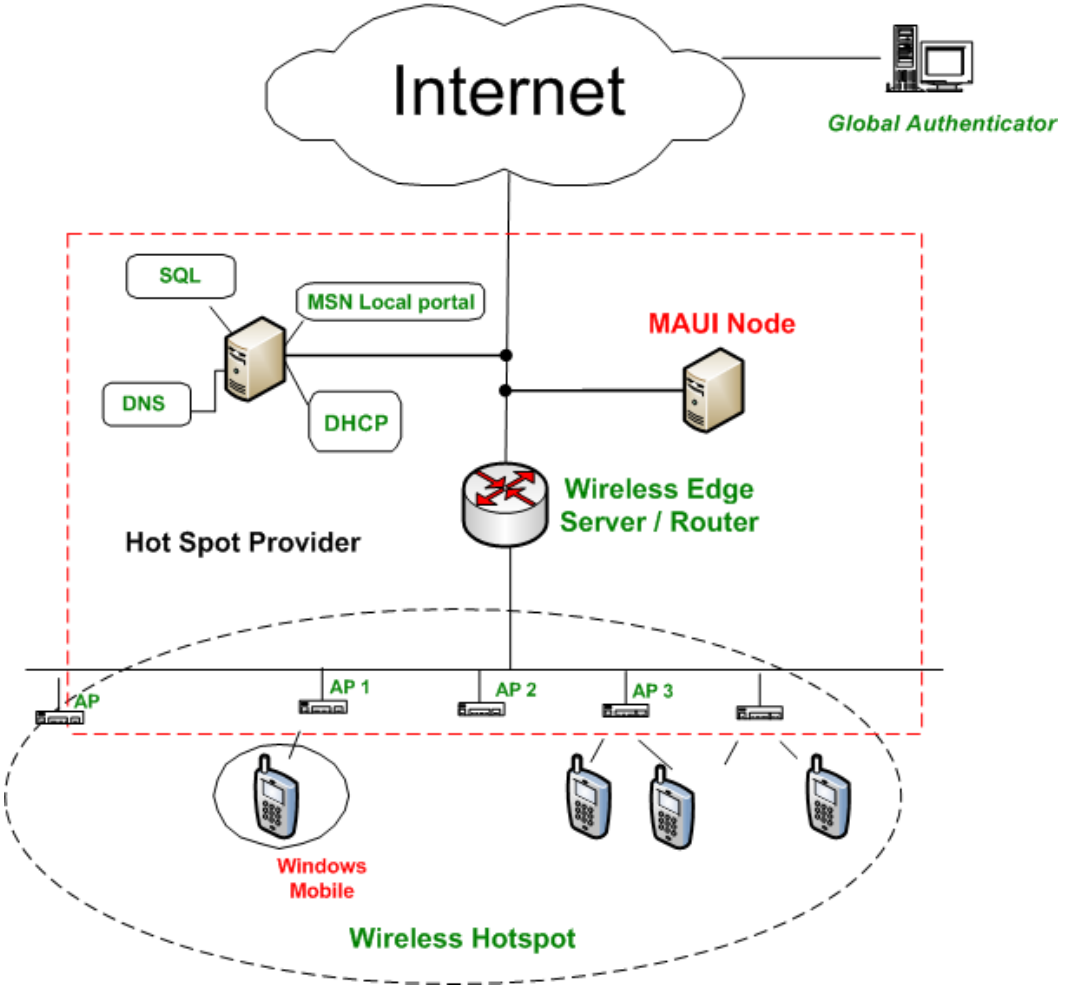
augment Wi-Fi hot spots or femtocells with cloudlets .

**advantages**

- does not use cellular spectrum
- short round-trip-times between mobile & cloud(let)
- optimal performance

**research challenges**

- Offload framework
- caching
- security & privacy



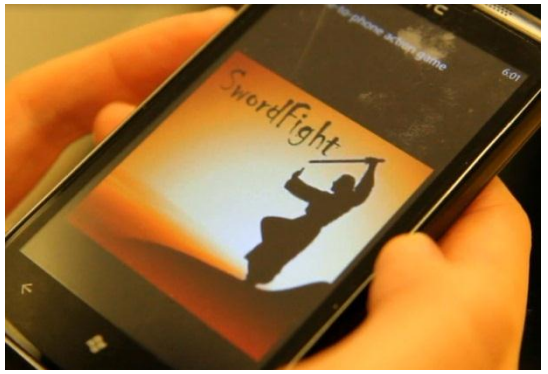
# … now for something different

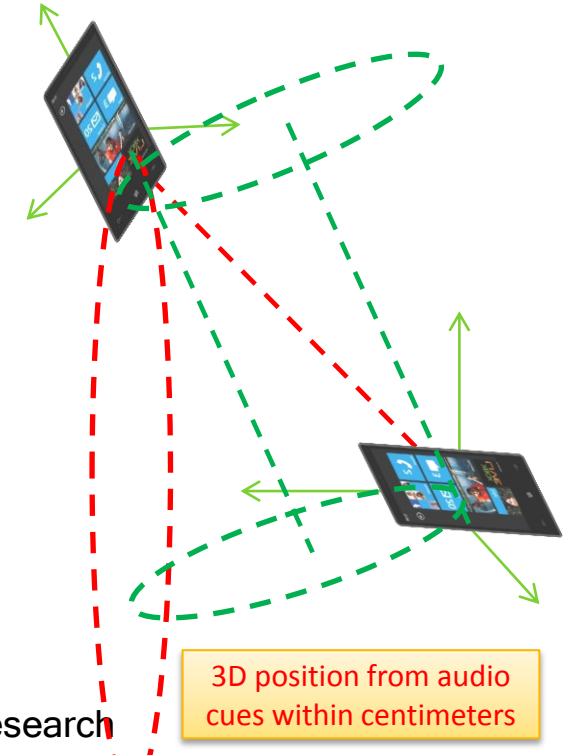
## what if a cloud does not exist?

# ad hoc multiplayer gaming

MobiSys 2012

## high resolution real-time continuous location




**HLPP**: a new class of games & applications

**H**igh-Speed, Real-Time, **L**ocational **P**hone-to-**P**hone

# Looking ahead….

- multi-player mobile gaming will continue to be a huge revenue generator
- new devices (e.g. HUD) will tax the networks & infra-structure even more…
- perennial challenges: bandwidth management, latency reduction, energy management, localization, etc. remain
- Prediction: there will be a large number of extensive cloud services for games

# what is the killer app you say…


# ….it’s killing time

# Thanks!