Mobile Performance from Radio Up

battery, latency, and bandwidth optimization for the wireless web

Ilya Grigorik igrigorik@google.com

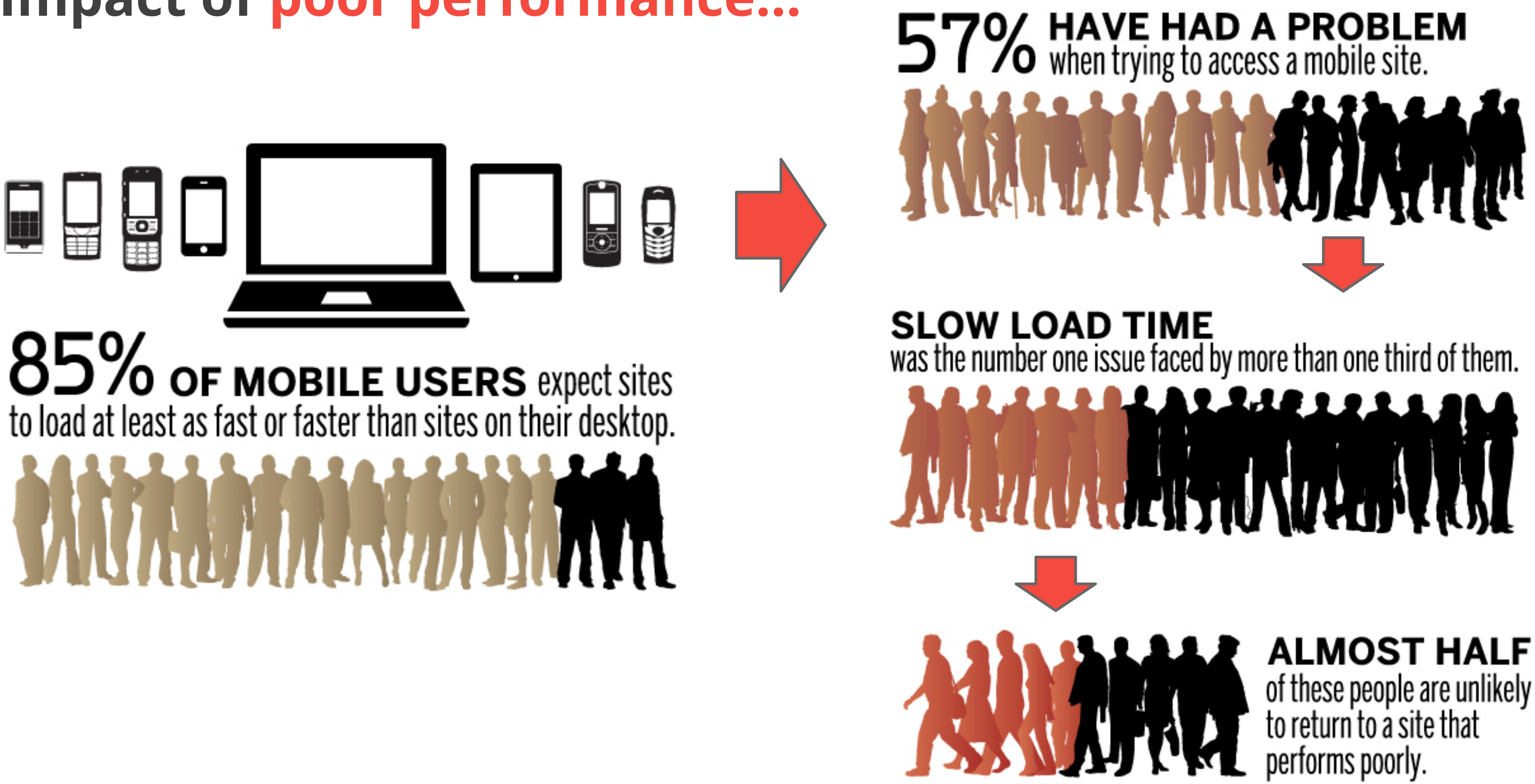
Video @ bit.ly/io-radioup

Wireless != Wired

- **Different** design constraints
- **Different** performance characteristics
- **Different** optimization criteria

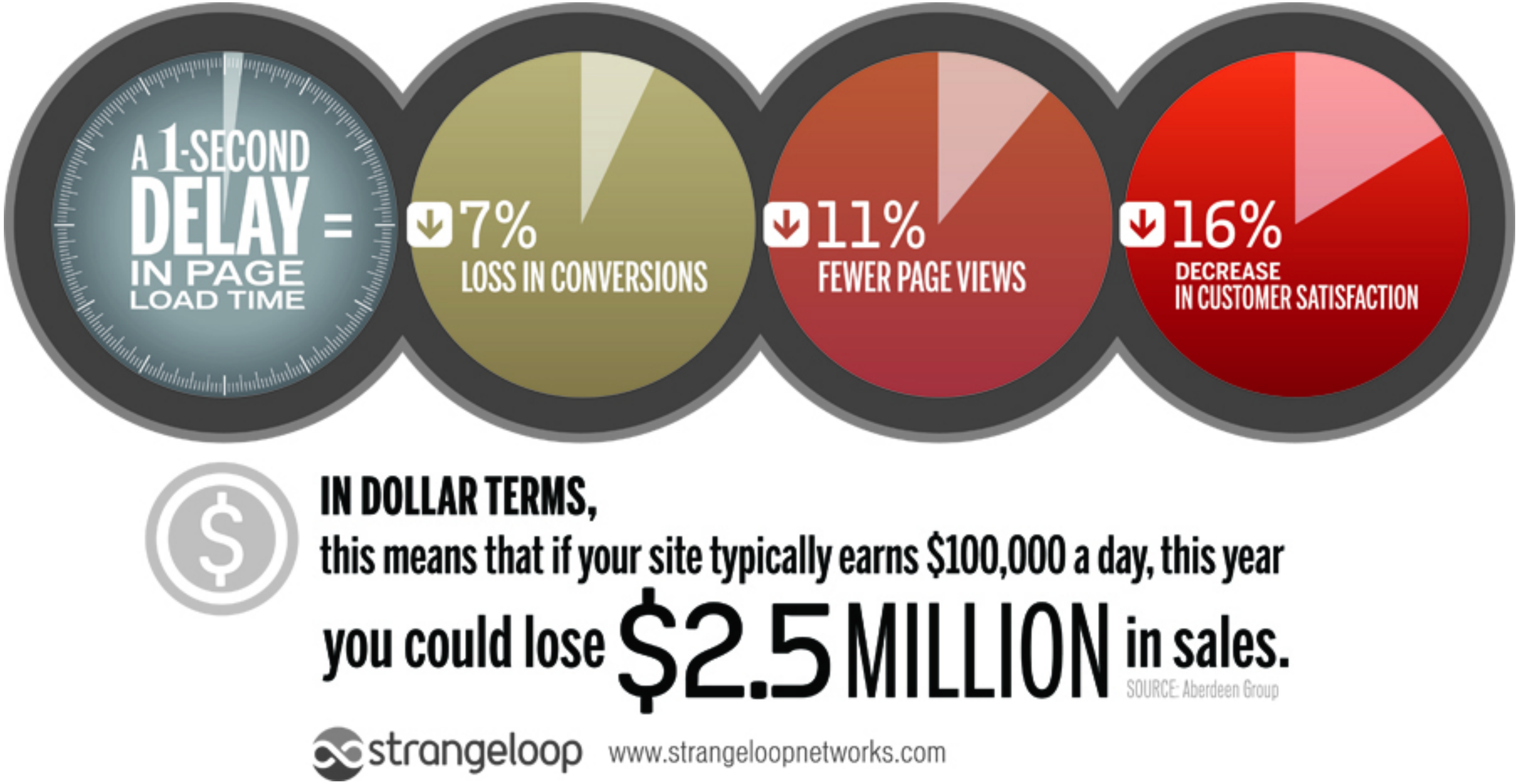
*If you continue treating wireless networks same as tethered, you will succeed at delivering a reliably **sub-par experience** to your users.*

Impact of poor performance...



[Impact of 1-second delay]() - Strangeloop

Impact of 1-second delay...

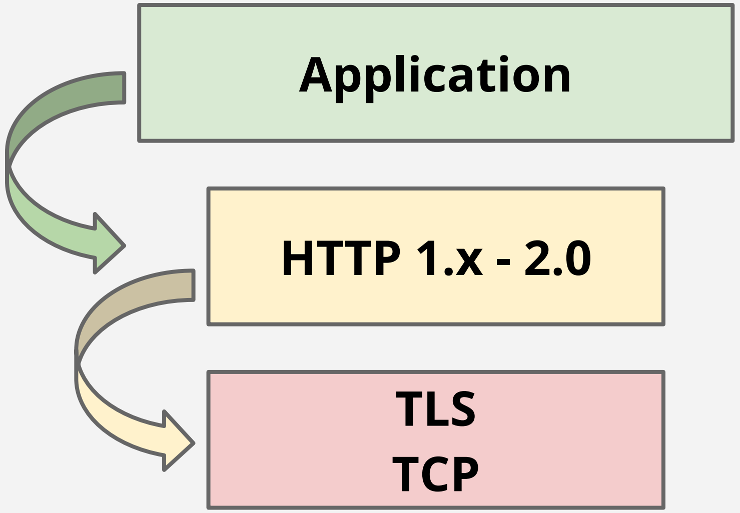


Impact of 1-second delay - Strangeloop

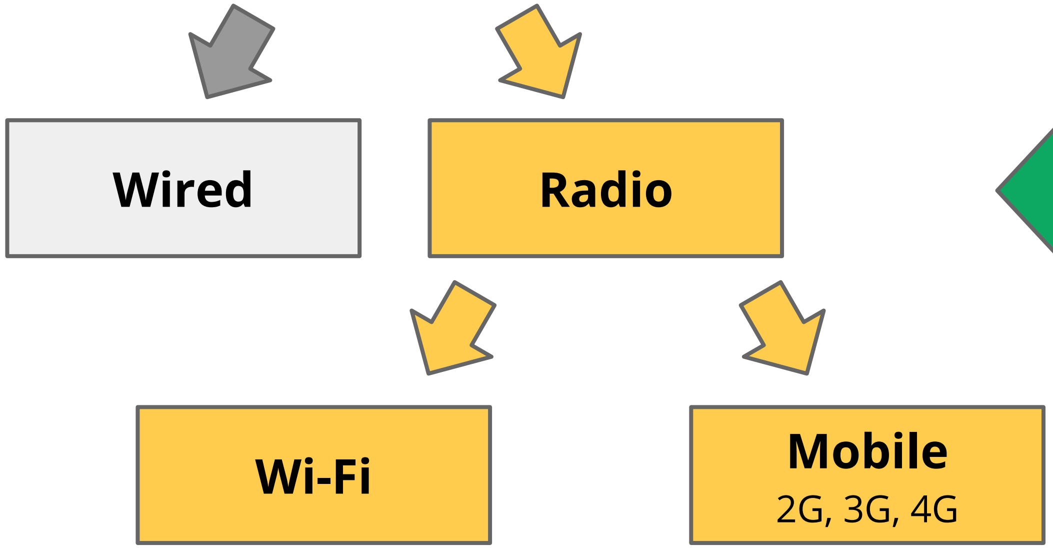
Our agenda today is...

1. **Radio performance 101**
    - Wi-Fi vs. Mobile
2. **Performance tips and recommendations**
3. ...
4. **Profit! ***

* Fill steps 3-n with your own awesome application sauce.

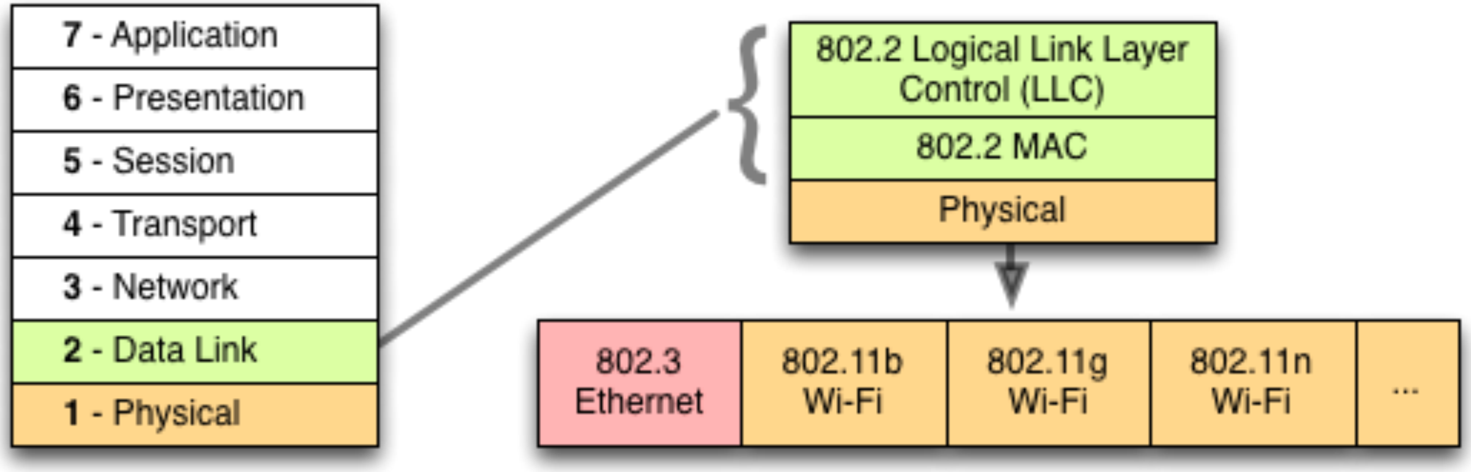


- Minify CSS, JavaScript and HTML
- Inline small images, CSS, and JavaScript
- CSS/JavaScript combining
- Domain Sharding
- JavaScript optimization
- GPU performance
- ....



- **Us today, right now... in fact!**
- Wi-Fi, 3G / 4G performance
- Battery life optimization
- Radio Resource Controller (RRC)

Wireless LAN, aka Wi-Fi...



- Wireless extension to existing LAN infrastructure
- Same protocol stack, new physical medium
- First commercial success ~1999 with 802.11b (11 Mbps)

- *Target: desktop, laptop*

Carrier Sense Multiple Access + Collision Detection

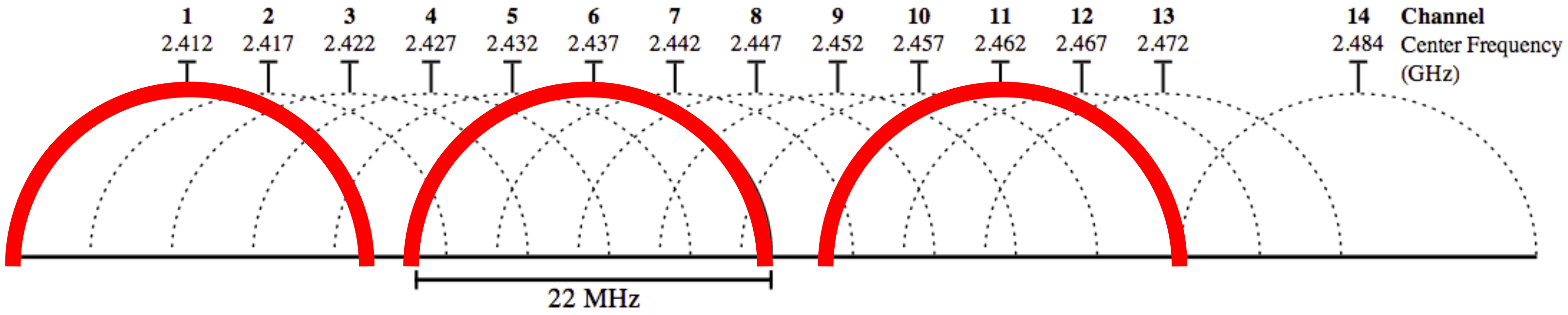
- **Is anyone talking?**
  - **No:** begin transmission
  - **Yes:** wait until they finish
    - **Collision:** stop, sleep for *rand()* with backoff, retry

Channel load must be kept below 10%.

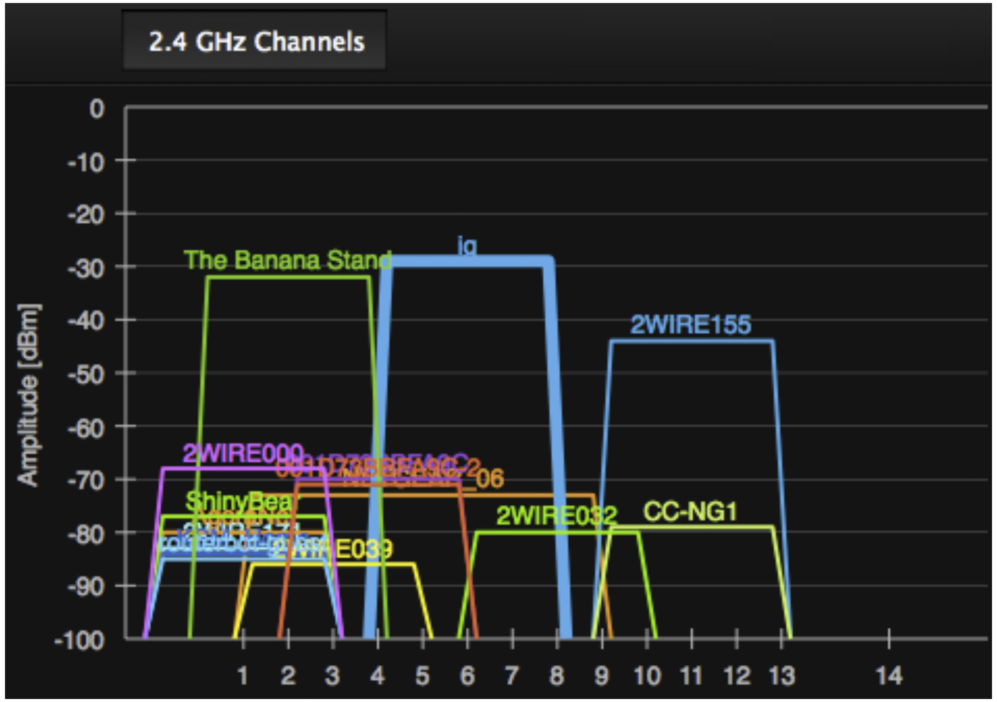
*"If the load increases, the number of collisions will quickly rise, leading to **unstable performance of the entire network**."*

[High Performance Browser Networking: Wi-Fi]

Wi-Fi channels 101



Non-overlapping channels: **1, 6, 11**

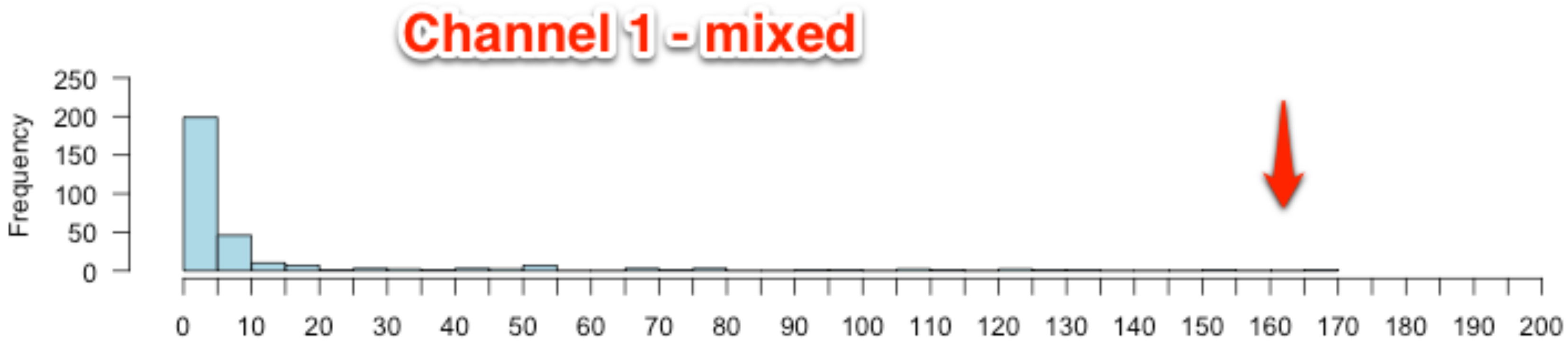


- Wi-Fi is a **victim** of its own success
- Any user, on any network (in the same channel) can and will affect your latency and throughput!
- 10-20+ overlapping networks in same channel
  - shared access medium

Real-world Wi-Fi performance: 2.4 Ghz vs 5 Ghz *(YMMV)*

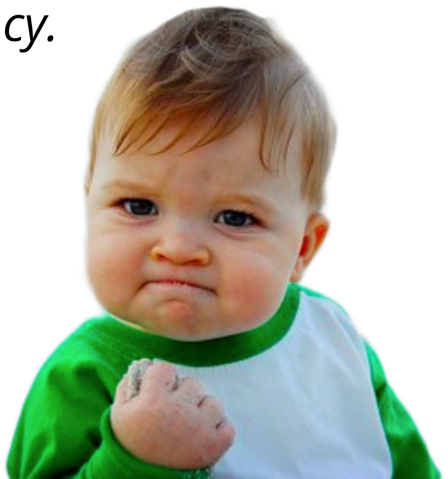
A	B	C	D	E	F	G	H	I	J
	25%	50%	75%	90%	95%	99%	100%	Median	
Channel 1 - N only	3.14	3.27	4.68	16.81	53.23	118.08	136.74	3.27	(new)
Channel 1 - mixed	3.19	3.59	6.72	37.25	69.62	125.94	167.44	3.59	
Channel 6 - mixed	3.30	4.11	8.51	48.98	89.81	138.20	189.20	4.11	(original)
Channel 11 - mixed	3.95	7.72	15.90	48.70	74.28	140.34	188.85	7.72	

Real-world 1st hop latency



	Median (ms)	95% (ms)	99% (ms)
2.4 Ghz	6.22	34.87	58.91
5 Ghz	0.90	1.58	7.89

*Upgraded router, removed **~50 ms** of latency.*



Sample data from my own home Wi-Fi network...

Adapt to variable bandwidth

- Adapt, do not predict!
- **Adaptive bitrate streaming**
  - 5-10 second chunks of video content

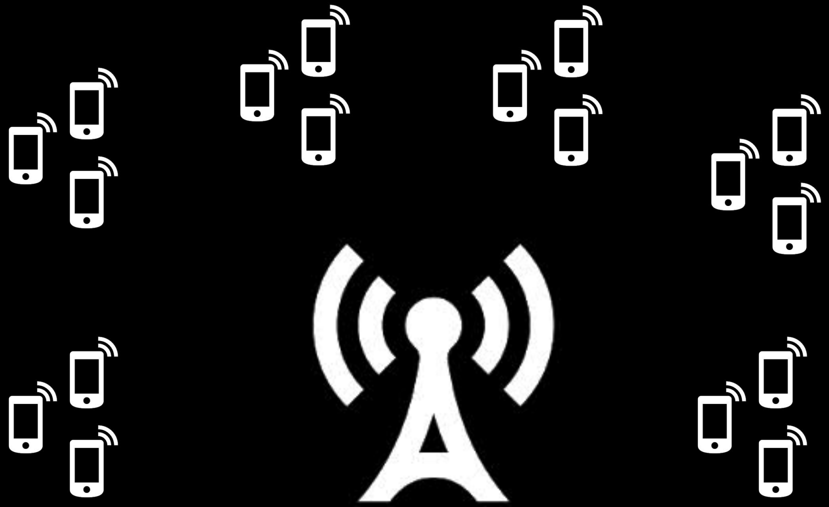
Adapt to variable latency and jitter

- High variability for first hop **for every packet**
  - Variability != packet loss
- Leverage WebRTC ... (check out the I/O session!)

2G, 3G, 4G...

*have **fundamentally different architecture** at the radio layer*

Design constraint #1: "Stable" performance + scalability



- **Control** over network performance and resource allocation
- Ability to manage **10~100's of active devices** within single cell
- Coverage of much larger area

Design constraint #2: Maximize battery life

- Radio is the **second most expensive** component (after screen)
- Limited amount of available power (as you are well aware)

Constraint #1: "Stable" performance + scalability

+

Constraint #2: Maximize battery life

=

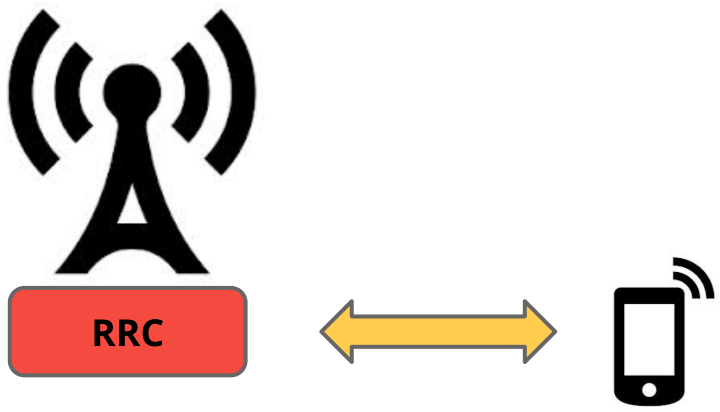
Radio Resource Controller (RRC)

Radio Resource Controller

- **Phone:** Hi, I want to transmit data, *please?*
- **RRC:** OK.
    - Transmit in [x-y] timeslots
    - Transmit with Z power
    - Transmit with Q modulation

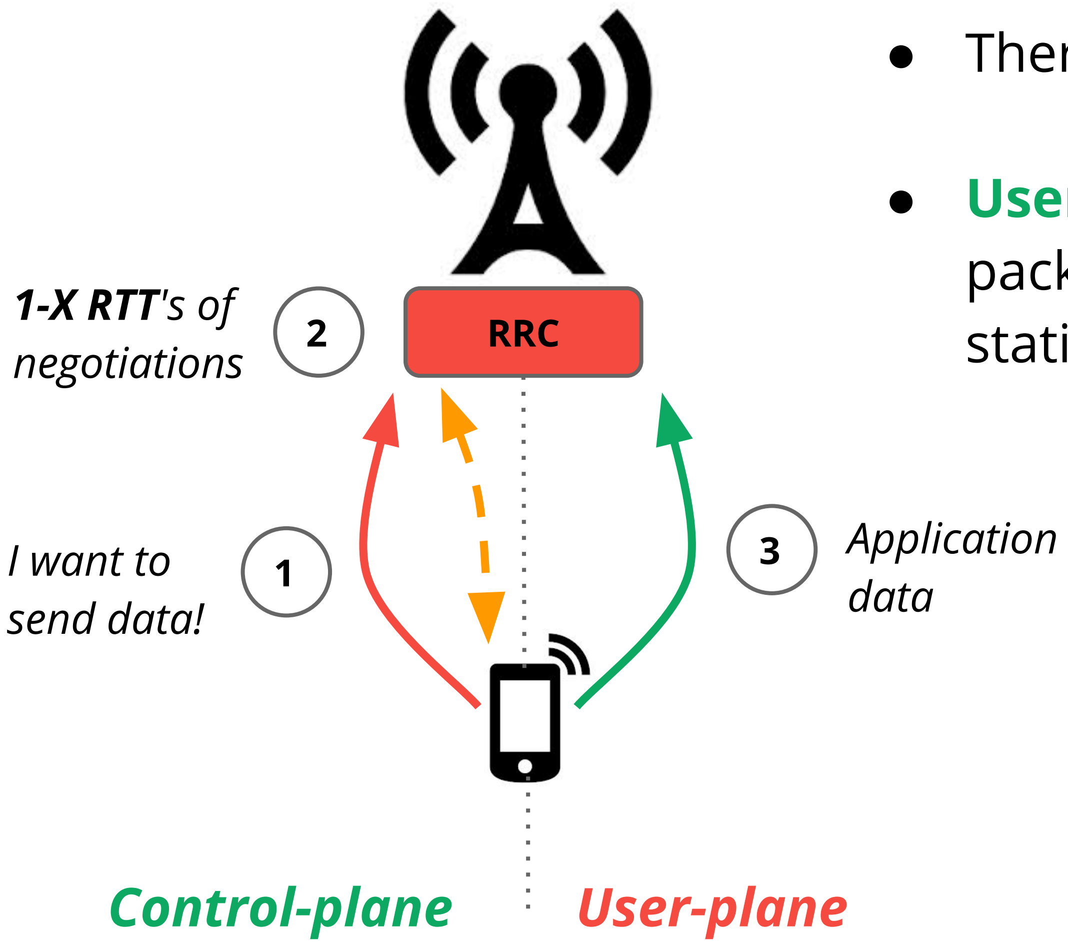
... (some time later) ...

- **RRC:** Go into low power state.



*All **communication and power management is centralized** and managed by the RRC.*

Control and User-plane latencies

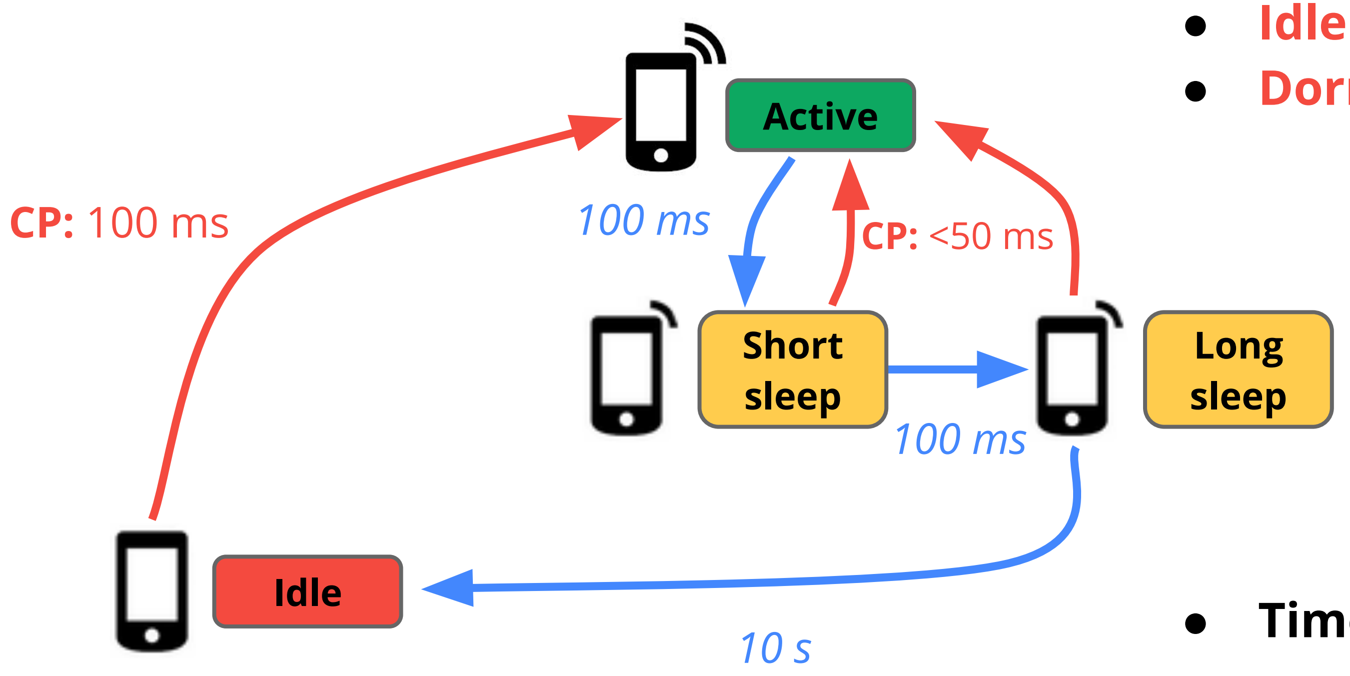


- There is a **one time** cost for **control-plane** negotiation
- **User-plane** latency is the **one-way** latency between packet availability in the device and packet at the base station

	LTE	HSPA+	3G
Idle to connected latency	< 100 ms	**< 100 ms**	**< 2.5 s**
User-plane latency	< 5 ms	< 10 ms	< 50 ms

Same process happens for incoming data, just reverse steps 1 and 2

LTE power state transitions

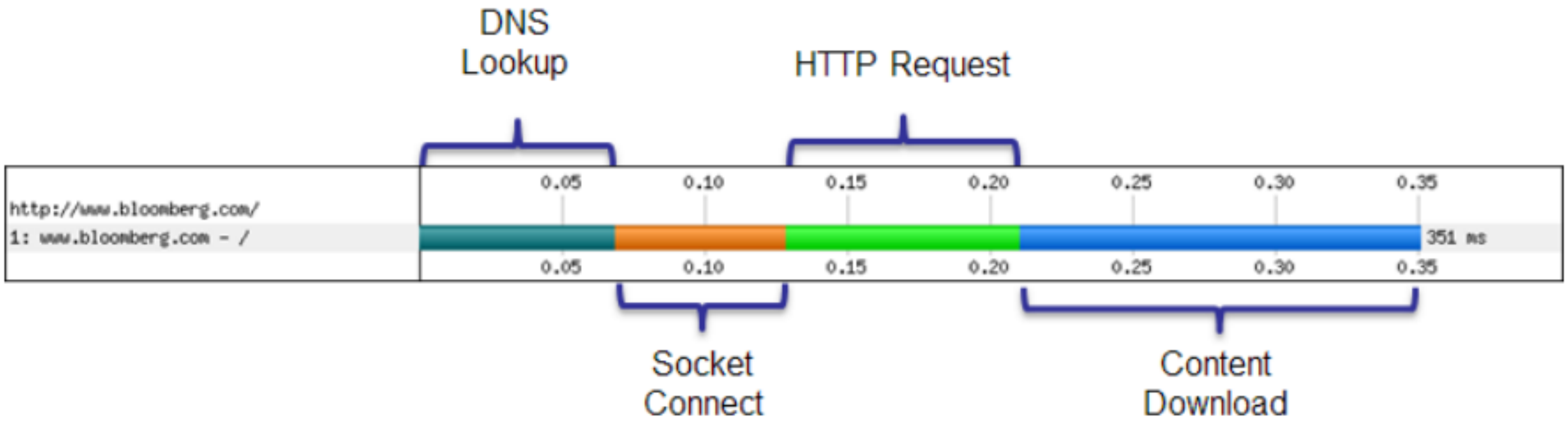


- **Idle to Active:** 100 ms control-plane latency
- **Dormant to Active:** 50 ms control-plane latency

- **Timeout driven** state transitions back to idle
  - 100 ms > 100 ms > 10 s > Idle

- Similar state machine for 3G devices
  - Except CP latencies are ***much higher***

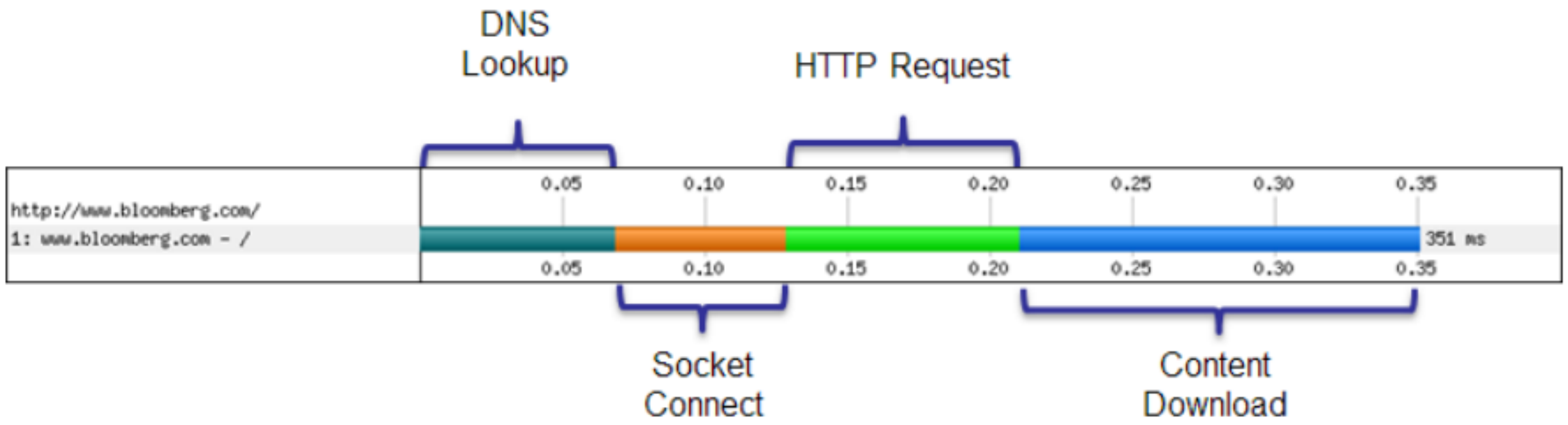
The *(short)* life of a web request



- *(Worst case)* **DNS lookup** to resolve the hostname to IP address
- *(Worst case)* **New TCP connection**, requiring a full roundtrip to the server
- *(Worst case)* **TLS handshake** with up to two extra server roundtrips!

- **HTTP request**, requiring a full roundtrip to the server
- **Server processing time**

The *(short)* life of a web request



	HSPA *(200 ms RTT)*	**HSPA+ / LTE** *(80 ms RTT)*
Control plane	***(200-2500 ms)***	***(50-100 ms)***
DNS lookup	200 ms	80 ms
TCP Connection	200 ms	80 ms
TLS handshake	*(200-400 ms)*	*(80-160 ms)*
HTTP request	200 ms	80 ms
Network latency overhead	**600 - 3500 ms**	**240 - 500 ms**

Explains a lot of the variability!

Network overhead of one HTTP request!

Decouple user feedback from network activity

Delay	User reaction
0 - 100 ms	Instant
100 - 300 ms	*Feels sluggish*
300 - 1000 ms	Machine is working...
1 s+	Mental context switch

Just the network overhead!

Acknowledge user input immediately

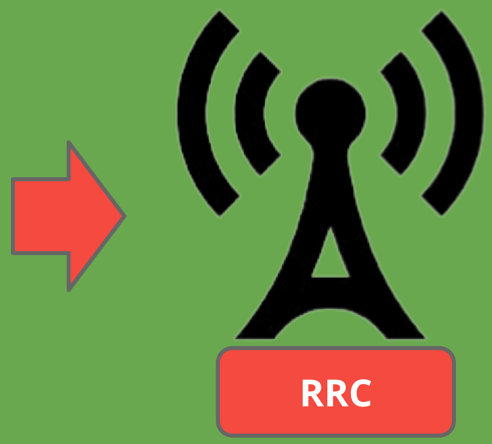
- *100-200 ms budget for instant feedback*

All communication should be asynchronous

- *provide feedback after 2s (progress bar or status update)*
- *provide a choice after 5s (wait, abort, retry?)*

Anticipate RRC latency...

	HSPA *(200 ms RTT)*	**HSPA+ / LTE** *(80 ms RTT)*
Control plane	**(200-2500 ms)**	**(50-100 ms)**
DNS lookup	200 ms	80 ms
...	...	...



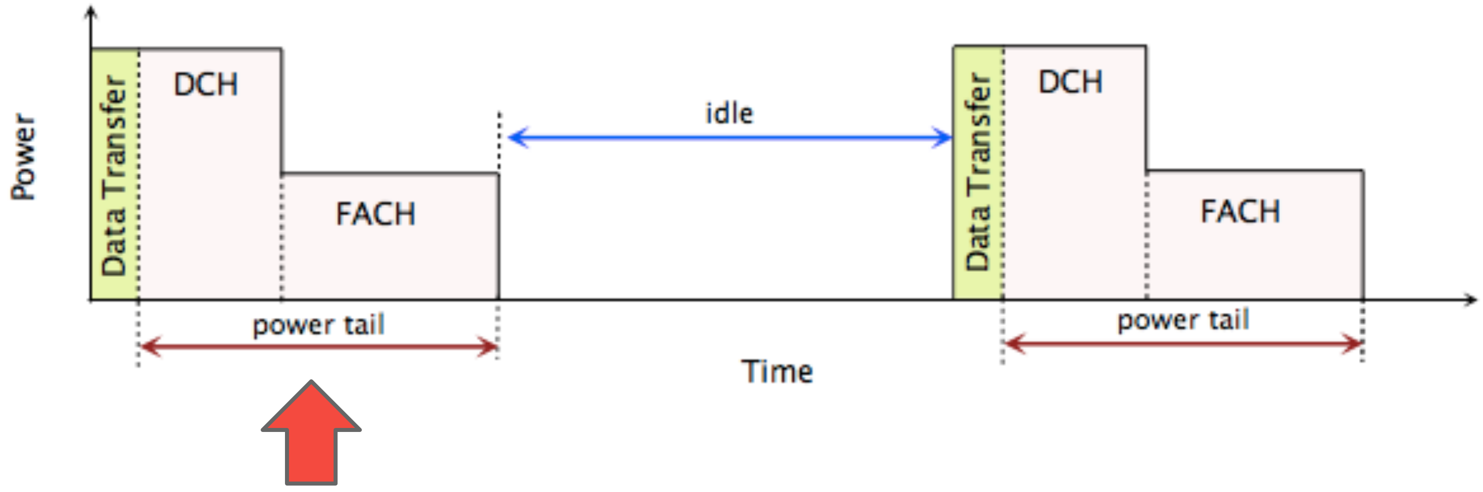
Plan for "first packet" delay

- *100's to 1000's of milliseconds of delay for first packet*
- *Adjust UX accordingly!*

Much of the "variability" is explained by RRC delays

- *you can't predict it, but assume you will incur it*

Watch those energy tails!



Wasted energy

3G state machine

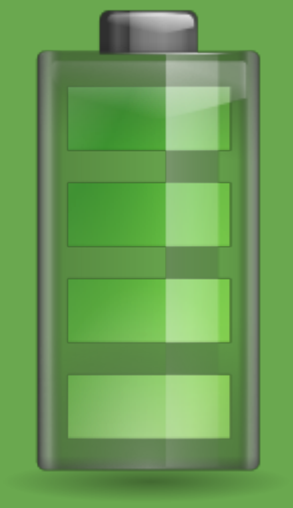
- **DCH** = Active
- **FACH** = Low power
- **IDLE** = ...

Every data transfer, both big and small, will:

- cycle radio into high power
- reset the power timeouts

Intermittent transfers are the most reliable way to **destroy the battery life**.

Hands on example....



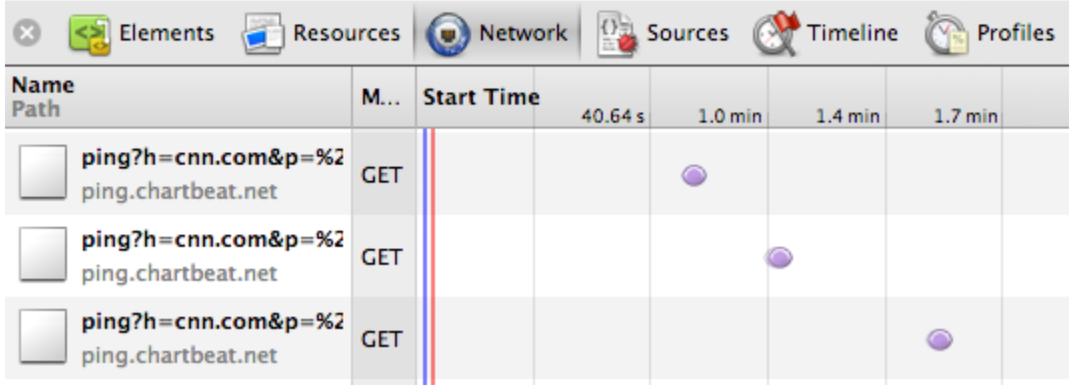
- **5 Wh** battery capacity
- 5 Wh * 3600 J/Wh = **18000 J battery capacity**

- **10 Joules** of consumed energy per "cycle"
  - *idle → high-power → low-power → idle*

- 60 minutes * 10 Joules = **600 Joules** of consumed energy **per hour**
- **3% of battery capacity per hour!** (600 J / 18000J)

Pandora beacons: 0.2% total bytes == 46% battery

Measuring **energy** & **radio**...



Ping, ping, ping, ... where'd my battery go?

Watch out for...

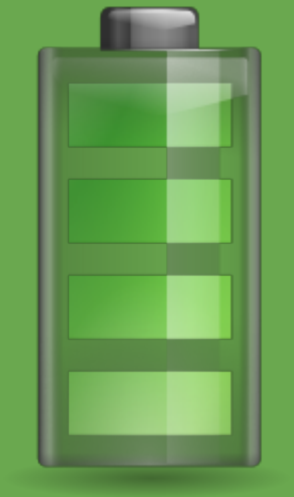
- *"real-time"* ***analytics***
- *"real-time"* ***comments***
- *"real-time" <widget>...*

Sidenote: ***not*** *an issue with Google Analytics real-time!*

- Record live trace on the phone, or import a pcap!
- Battery + radio models for 3G and 4G
- Performance linter... caching, compression, etc.
- ...

It's all about the battery...

- Radio is the **second most expensive** (energy) component
- Radio use at full power can **drain full battery in hours**
- Radio use is **expensive regardless of transfer size**



Prefetch data

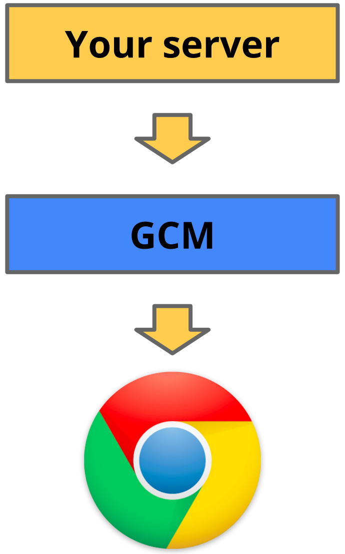
- *turn off the radio, keep it idle*

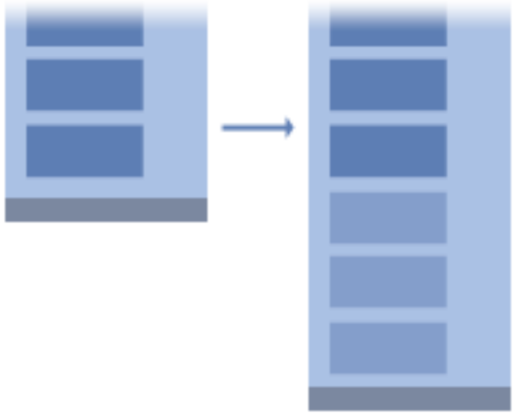
Minimize periodic data transfers

- *avoid polling, use adaptive strategy*
- *leverage server push*
- *coalesce requests, defer requests*

Prefetch data, leverage *(smart)* push...

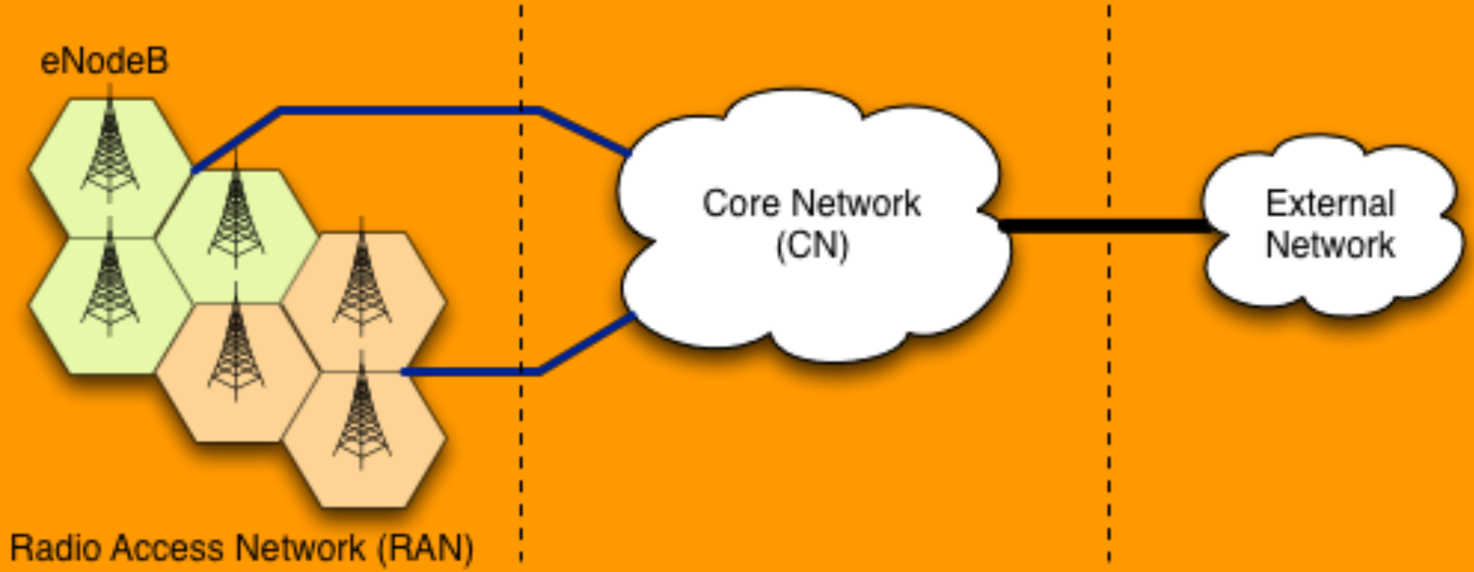
- Google Cloud Messaging for **Chrome** (new!)
- Google Cloud Messaging for **Android**
    - *collapse_key, delay_while_idle, time_to_live*





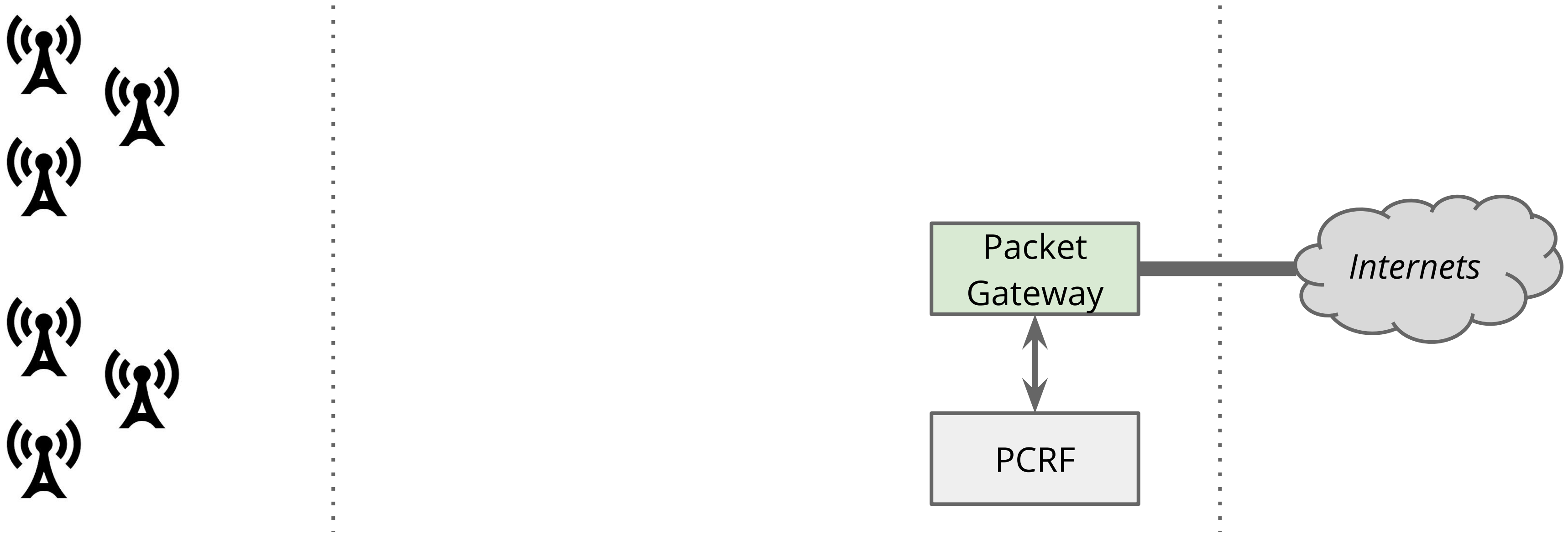
- **Prefer prefetch** *vs.* on-demand
    - *how much to prefetch? measure.*
- **Streaming data?**
    - *download in one shot where possible*

GCM for Chrome, GCM for Android



We've covered the basics of the RAN, now (for fun) let's take a look at the ***Core Network architecture****...*

Packet Gateway



Packet Gateway terminates all connections

- IP assignment is done at the PGN
- PGN acts as a NAT

Physical layer connectivity != Application connectivity

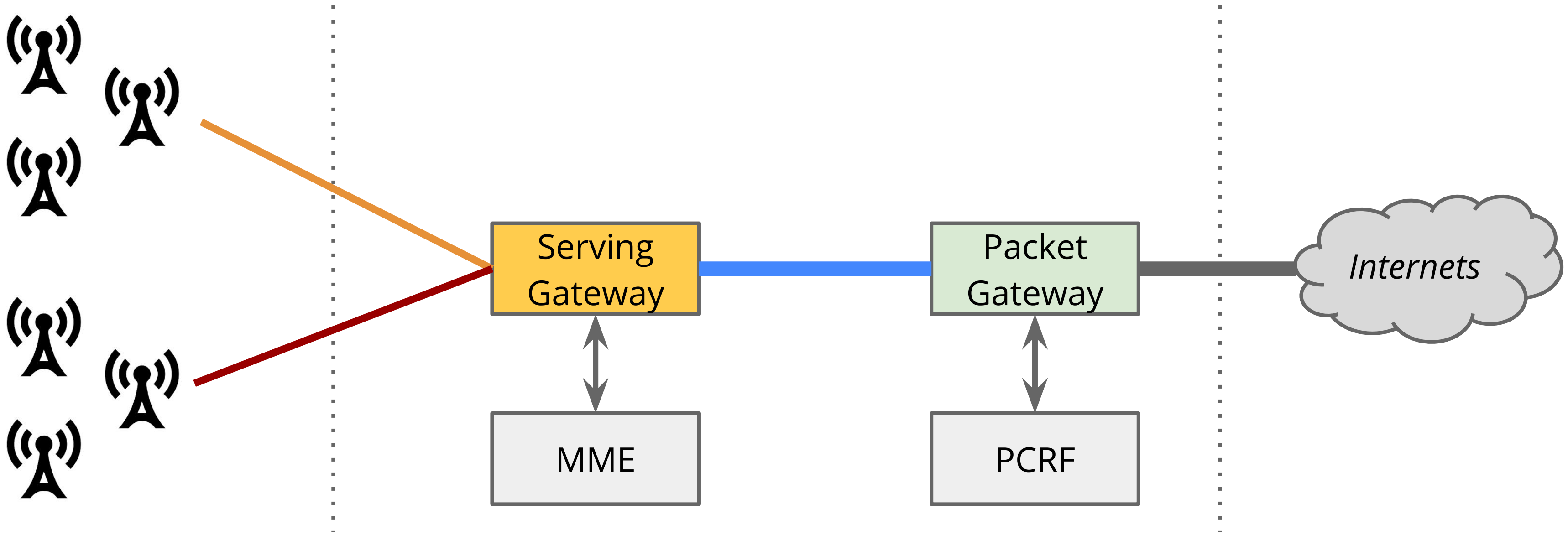
... turning off the radio **does not close** your TCP connection!

```
window.setInterval("keepSessionAlive()", 10000);
```

Carrier timeouts: 5-30 minutes!

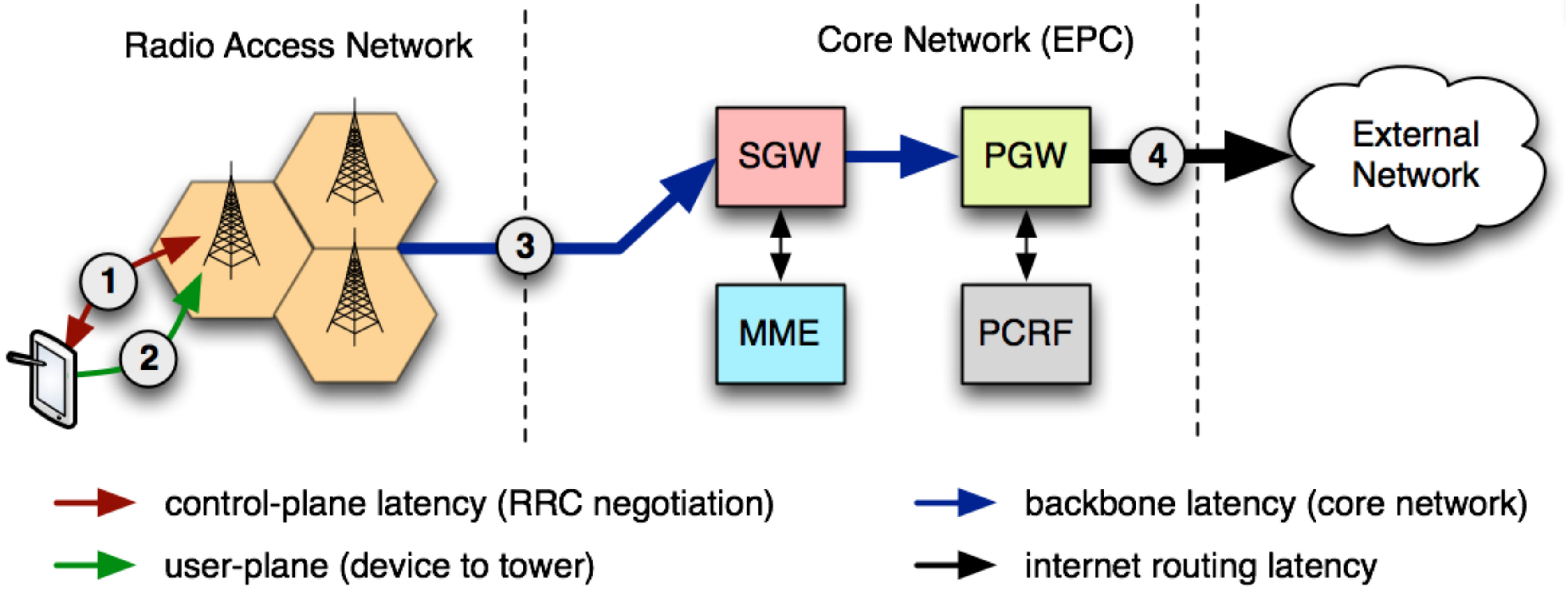
- *skip the "I'm alive" beacons, please...*
- *are you sure it's not your own servers forcing timeouts? :)*

Serving Gateway



- **Serving Gateway (SGN)** is the **mobility anchor**
  - Towers are grouped into "tracking areas"
  - SGN may not know which tower the user is in!
- **Mobility Management Entity** (MME)
  - The "user database" for the carrier
  - Billing status, enabled features, ...
  - *Location of the user in the network!*

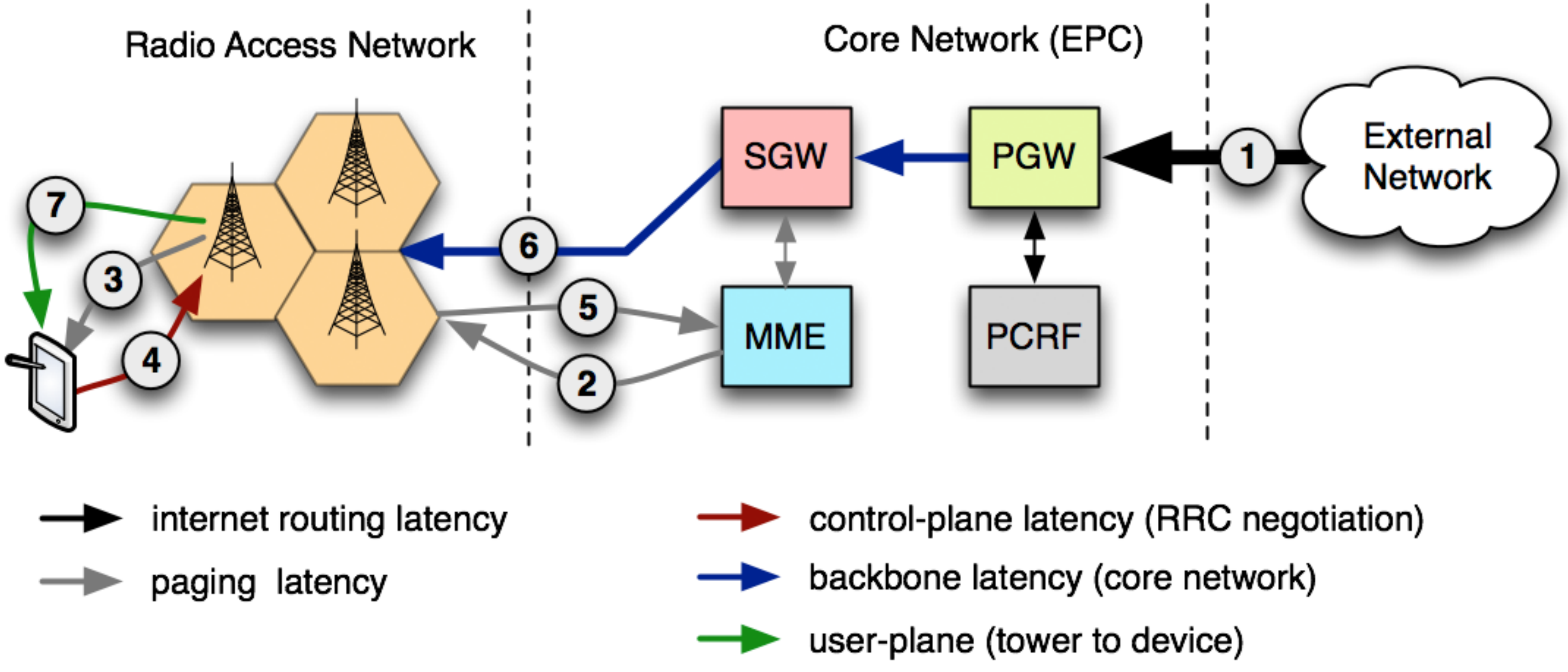
Outbound data-flow



	LTE	HSPA+	HSPA	EDGE	GPRS
Core network (AT&T)	40-50 ms	50-200 ms	150-400 ms	600-750 ms	600-750 ms

** highly variable between carriers, local infrastructure, local wireless weather….*

Inbound data-flow

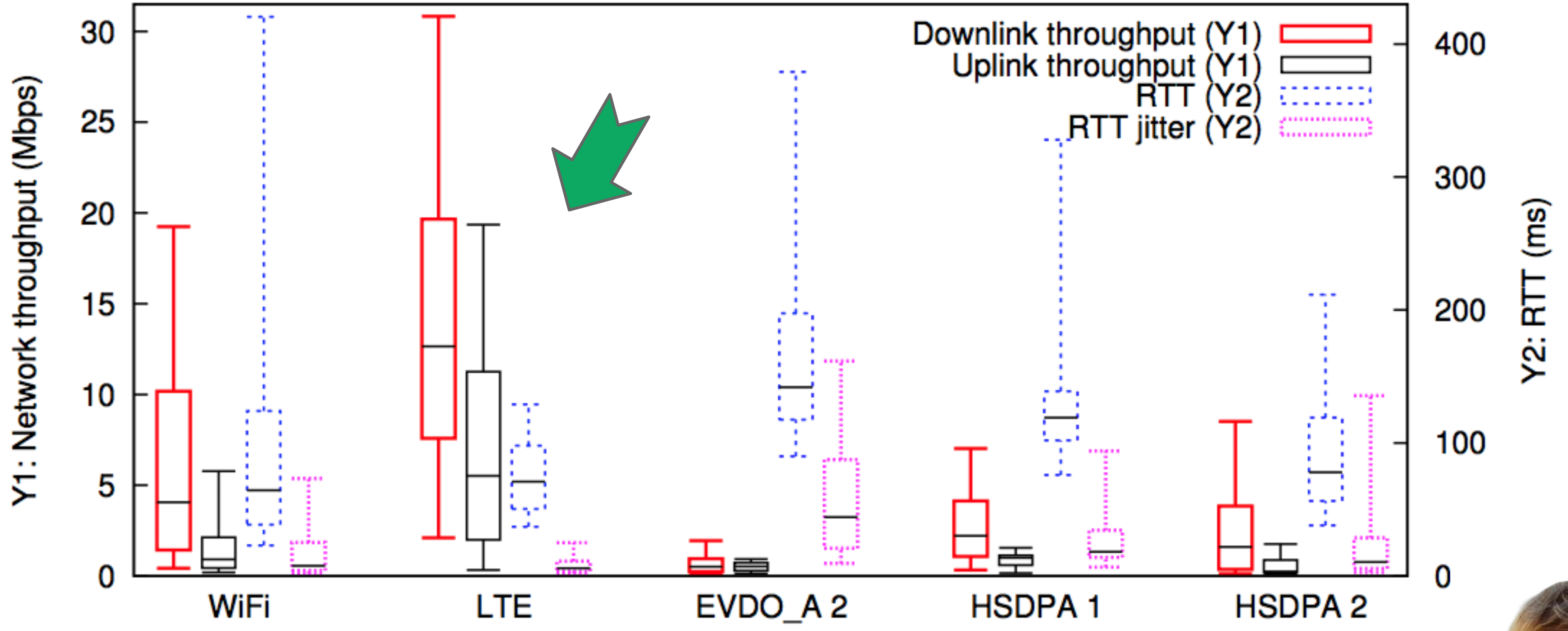


	LTE	HSPA+	HSPA	EDGE	GPRS
Core network (AT&T)	40-50 ms	50-200 ms	150-400 ms	600-750 ms	600-750 ms

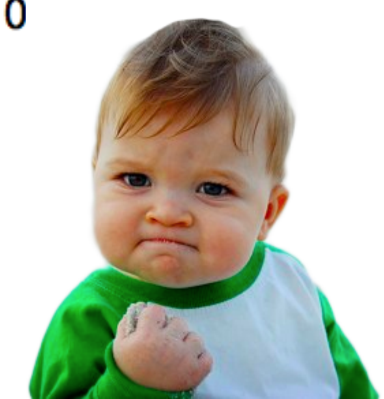
** highly variable between carriers, local infrastructure, local wireless weather….*

*really... **all that**, for a single TCP packet?*

Good news everybody! ....



- ***LTE shows better performance profile across the board!***
  - *perhaps all of that complexity will pay off after all...*

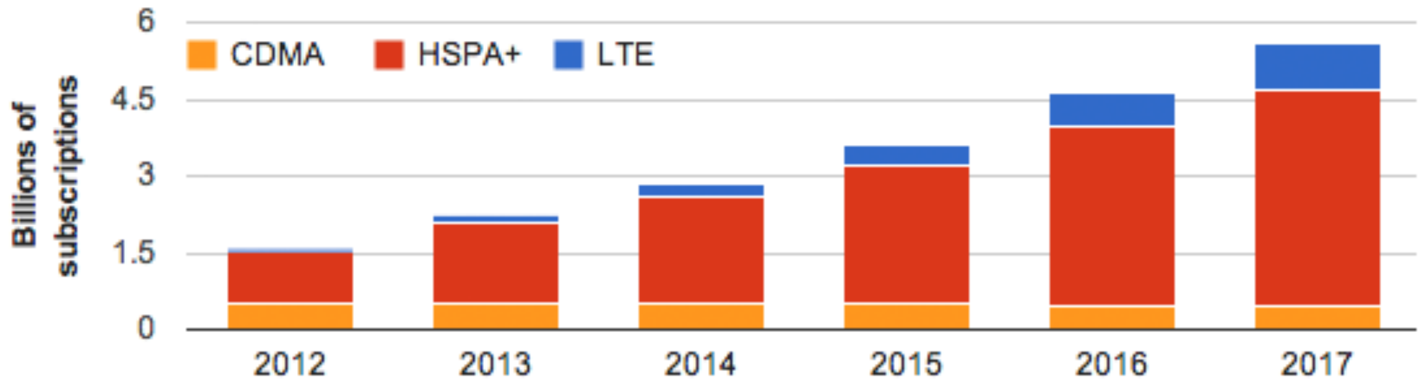


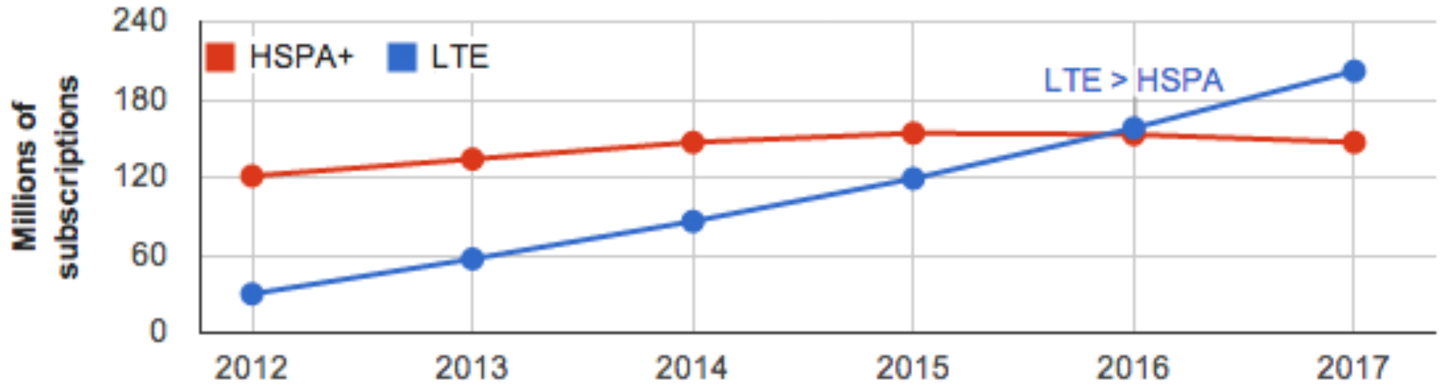
[2011, 3300 users, 14,000 runs (MLab)]

Burst your data and return to idle!

- Mobile radio is **optimized for bursty data transfers**
    - *Bandwidth / latency estimation is a recipe for trouble...*
- **Group requests** together, download as much as possible
    - *For streaming transfers, consider prompting Wi-Fi switch...*

Not so good news everybody! ....





HSPA+ will be the dominant network type of the next decade!

- *Latest* HSPA+ releases are comparable to LTE in performance
- 3G networks will be with us for ***at least*** another decade
- LTE adoption in US and Canada is ***way ahead*** of the world-wide trends

[4G Americas - Statistics]

Design for variable network performance & availability

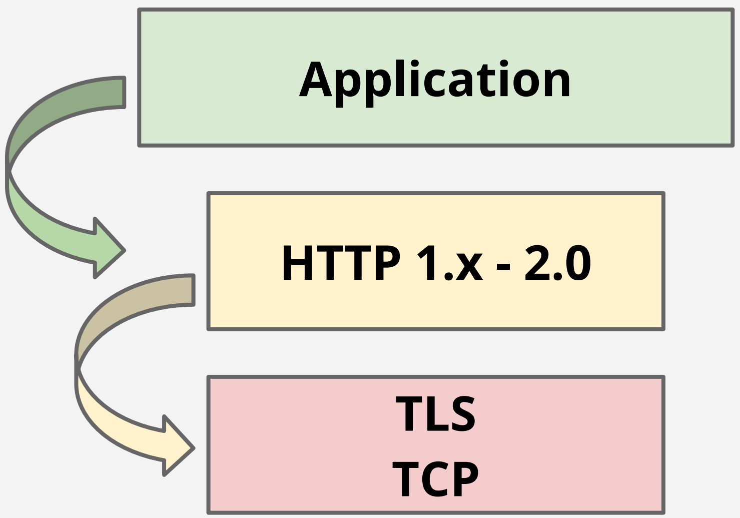
- It's a **multi-generation future**: 2G, 3G, 4G
  - *users migrate between G's all the time... **plan for it!***
- Bandwidth and latency is **highly variable**
  - *burst your data, and return to idle...*
- **Connectivity is intermittent**, errors will happen!
  - *have an offline mode - i.e. use a local cache*
  - *use a smart backoff algorithm, please!*

```
var backoff = backoff.strategy({
    randomisationFactor: 0,
    initialDelay: 60,
    maxDelay: 600
});

backoff.failAfter(10);
```

Apply application best practices

- **Measure** performance: RUM, synthetic, benchmarking
- **Eliminate** unnecessary resources
- **Optimize** rendering and JavaScript performance
- ...



- Minify CSS, JavaScript and HTML
- Inline small images, CSS, and JavaScript
- CSS/JavaScript combining
- Domain Sharding
- JavaScript optimization
- GPU performance
- ....

- **Fastest request is a request not made!**
- **Bytes are expensive** (literally.. $1+ MB)
  - *Android: public boolean isActiveNetworkMetered()*
- **Reduce latency:** use a CDN, use SPDY!
- **Cache data on the client**
  - *no really, cache the data! check your code.*
- **GZIP resources**
  - *no really, you would be surprised how many forget...*
- **Pick the appropriate image format**
  - *60% of bytes are images*
  - *Use lossy compression, check out WebP!*

Apply TCP, TLS, mobile and HTTP best practices...

</shameless self promotion>

- *Optimizing **TCP** server stacks*
- *Optimizing **TLS** deployments*
- *Optimizing for **wireless networks***
- *Optimizing for **HTTP 1.x**...*
- *Leveraging **HTTP 2.0 and SPDY**!*
- ...

~~$29.99~~ **Read Online for Free**

Brought to you by Velocity Conference

http://bit.ly/io-hpbn

Video @ bit.ly/io-radioup

Fin. Questions?

Book @ bit.ly/io-hpbn

+Ilya Grigorik
igrigorik@google.com

Google Developers