KAAZING

How I Learned to Stop Worrying & Love the λ
or
Dr. Strange-λ

Todd L. Montgomery
@toddlmontgomery

COLUMBIA
PICTURES
CORPORATION
PRESENTS

A

STANLEY
KUBRICK
PRODUCTION

λ

Haskell

λ

Erlang

Haskell

Clojure

Groovy

Erlang

Scala

Haskell

Clojure

C#

λ

Groovy

Erlang

C++11

Scala

Haskell

Clojure

C#

C11

λ

Groovy

Erlang

C++11

Scala


“Mr. President,
we must not allooooow a
Lambda Gap!”

Java 8

What could we do…

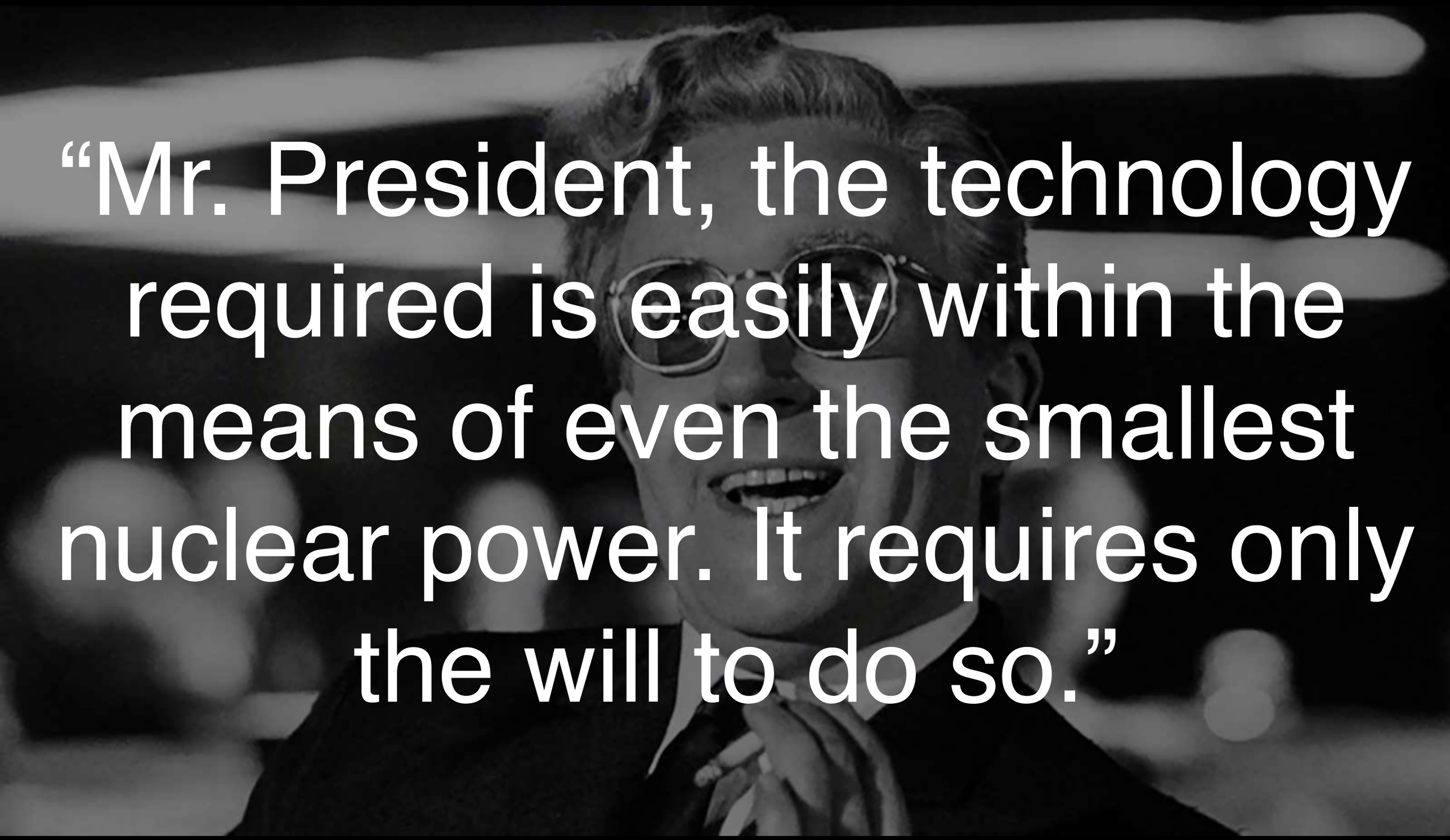
green field…

Perfect Timing!

Aeron

https://github.com/real-logic/Aeron

Low Latency
High Throughput
Open Source
Messaging Transport


“Mr. President, the technology required is easily within the means of even the smallest nuclear power. It requires only the will to do so.”

The Aeron Team

Todd Montgomery

Richard Warburton

Martin Thompson

Why?

In finance & other places…

Latency is [lost] opportunity

Capital Markets

The FX Trader

Currency Pairs

$ / £ = 0.6375
£ / ¥ = 123.77
¥ / $ = 0.0127

Trades

$1,000,000 = £637,500

£637,500 = 78,903,375¥

78,903,375¥ = $1,002,072

CHA-CHING!

78,903,375¥ = $994,182.53 !!!

¥ / $ = 0.0126

Ad Market

Sports & Entertainment

Logistics

Why Java?

Why even an OS?
Why even …


Do epic shit,
or die trying.


NUCLEAR WARHEAD
HANDLE WITH CARE
HI THERE!
NUCLEAR WARHEAD
HANDLE WITH CARE
DEAR JOHN

Design Principles

1. Garbage free in steady state running
2. Apply Smart Batching in the message path
3. Wait-free algorithms in the message path
4. Non-blocking IO in the message path
5. No exceptional cases in the message path
6. Apply the Single Writer Principle
7. Prefer unshared state
8. Avoid unnecessary data copies

https://github.com/real-logic/Aeron/wiki/Design-Principles

And, yes, **`Unsafe`**


USAF

But…

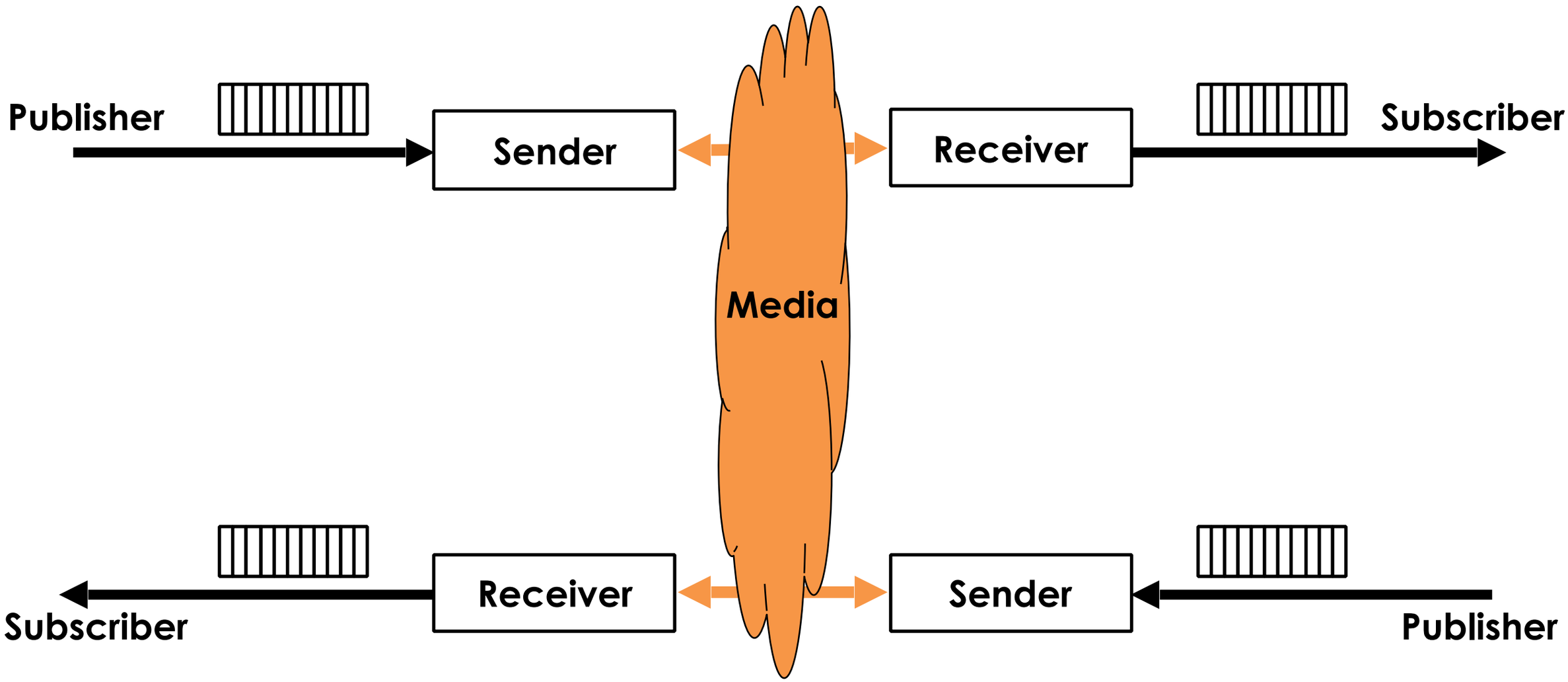
We have the launch codes

so, it’s cool

Architecture

— IPC Log Buffer

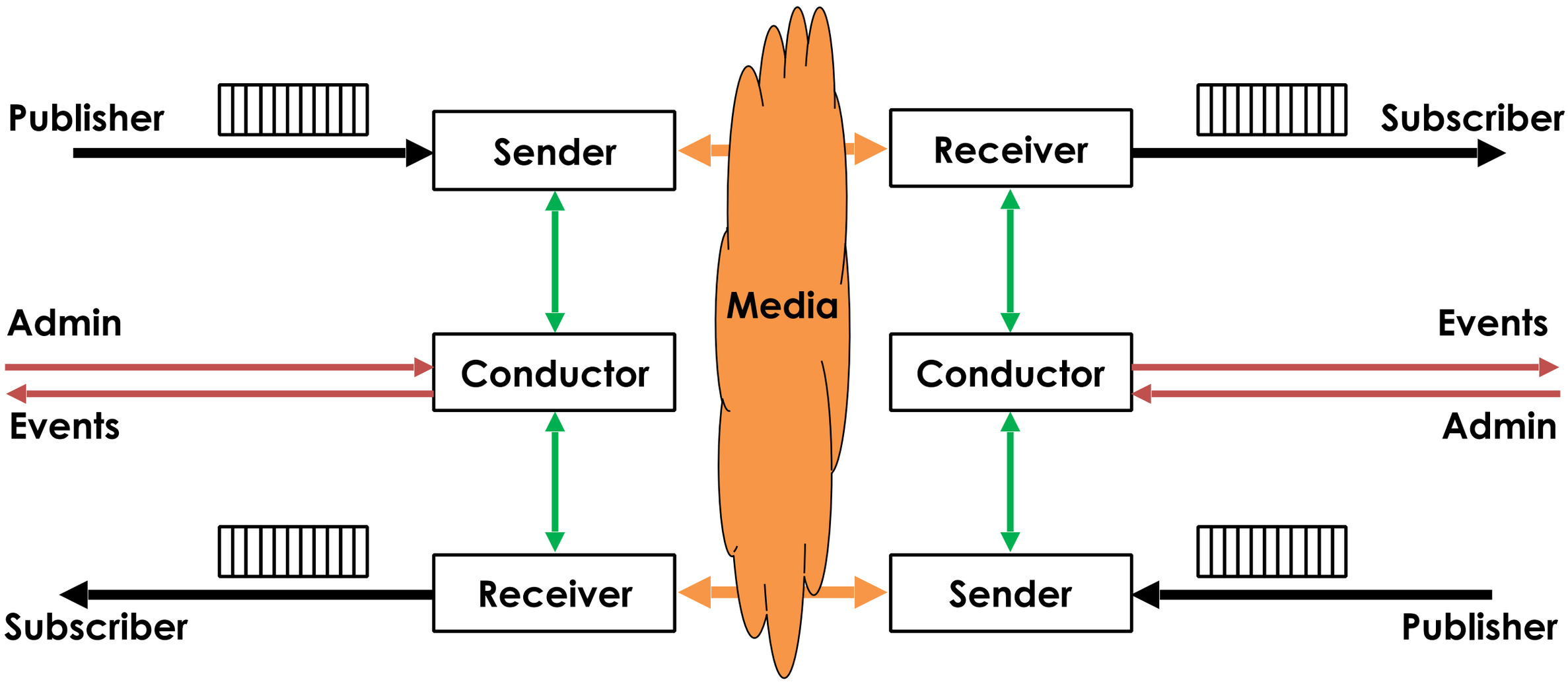
Architecture



IPC Log Buffer

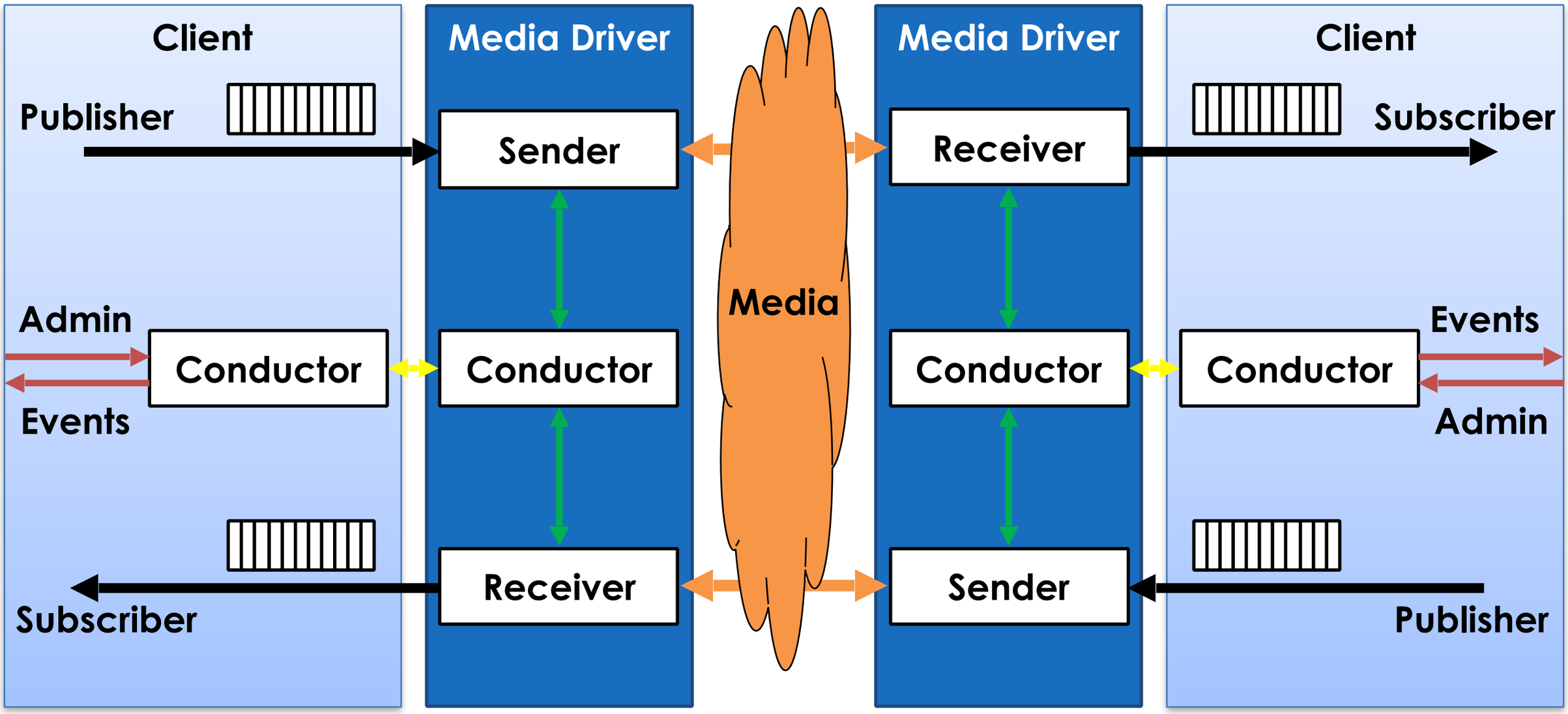
Media (UDP, InfiniBand, PCI-e 3.0)

Architecture



- IPC Log Buffer
- Media (UDP, InfiniBand, PCI-e 3.0)
- Function/Method Call
- Volatile Fields & Queues

Architecture



- IPC Log Buffer
- Media (UDP, InfiniBand, PCI-e 3.0)
- Function/Method Call
- Volatile Fields & Queues
- IPC Ring/Broadcast Buffer

Much, much more…

but…

Java 8…

What have we learned?

~~Stream API~~

Too much garbage!
Immature for now…

JMC

Very handy. Use it!

```
AtomicLong.getAndIncrement()
Unsafe.getAndAddLong(…)
```

LOCK XADD

Replacing a CAS on x86!! HOT!!

And, of course…

Lambdas [& Allocation]

Design Principles

1. **Garbage free in steady state running**
2. Apply Smart Batching in the message path
3. **Wait-free algorithms in the message path**
4. Non-blocking IO in the message path
5. No exceptional cases in the message path
6. Apply the Single Writer Principle
7. Prefer unshared state
8. Avoid unnecessary data copies

https://github.com/real-logic/Aeron/wiki/Design-Principles

```
public void something()
{
    List<Object> l = new ArrayList<>();
    final int value = 10;

    l.forEach(this::func);
    l.forEach((o) -> staticMethod(value));
    l.forEach((o) -> this.func());
}
```

```
public void something()
{
     List<Object> l = new ArrayList<>();
     final int value = 10;

     l.forEach(this::func);
     l.forEach((o) -> staticMethod(value));
     l.forEach((o) -> this.func());
}
```

Each time **`something`** called, it allocates 3 lambdas!!!

http://mail.openjdk.java.net/pipermail/lambda-dev/2014-August/012097.html

http://www.infoq.com/articles/Java-8-Lambdas-A-Peek-Under-the-Hood

```
public void something()
{
     List<Object> l = new ArrayList<>();
     final int value = 10;

     l.forEach(this::func);
     l.forEach((o) -> staticMethod(value));
     l.forEach((o) -> this.func());
}
```

Why? Capturing **`this`** or **`value`**

http://mail.openjdk.java.net/pipermail/lambda-dev/2014-August/012097.html

http://www.infoq.com/articles/Java-8-Lambdas-A-Peek-Under-the-Hood

But…

Lambdas Optimize Well

More things about Good Ol’ Java…

(not Java 8 specific)

Lack of Unsigned Types

Lack of [Efficient] Primitive Maps

```
Map<int,? extends Object>
Map<long,? extends Object>
```

http://openjdk.java.net/jeps/8046267

```
public static void main(final String[] args)
{
    byte a = 0b0000_0001;
    byte b = 0b0000_0010;

    byte flags = a | b;

    System.out.printf(
        "flags=%s\n",
        Integer.toBinaryString(flags));
}
```

```
public static void main(final String[] args)
{
    byte a = 0b0000_0001;
    byte b = 0b0000_0010;

    byte flags = a | b;

    System.out.printf(
        "flags=%s\n",
        Integer.toBinaryString(flags));
}
```

```
public void main(final String[] args)
{
    byte a = 0b0000_0001;
    byte b = 0b0000_0010;

    byte flags = a | b;

    System.out.printf(
        "flags=%s%n",
        Integer.toBinaryString(flags));
}
```

Error:(8, 24) java: incompatible types: possible lossy conversion from int to byte

Does not inspire confidence…

NIO

To an old network hacker, feels a bit “phoned” in…

`ByteBuffer` & “Strangs”

```
public void send(final String s)
{
    // b be reused ByteBuffer

    // allocating hot mess
    b.put(s.getBytes());

    // what could be… but JDK playin…
    b.put(s);
}
```

Allocate mutable from immutable…

http://mail.openjdk.java.net/pipermail/core-libs-dev/2014-September/028837.html

```
public String receive(final ByteBuffer b)
{
    // could reuse, but that's HARD
    byte[] tmp = new byte[b.remaining()];

    // allocating hot mess
    b.get(tmp, offset, length);
    return new String(tmp);

    // what could be… but JDK playin…
    return new String(b, offset, length);
}
```

Yes, **`Charset`**, yes…

http://mail.openjdk.java.net/pipermail/core-libs-dev/2014-September/028837.html


RADIOACTIVE III
CONTENTS.....
ACTIVITY.....
STRINGS!
TRANSPORT INDEX
7
UN CLASS 7

NIO & Locks

When Threads Collide…

“You can’t fight in here, this is the War Room!”

DatagramChannelImpl

```
public int write(ByteBuffer buf)
{
    synchronized (writeLock) {
        synchronized (stateLock) { … } … }
}

public int read(ByteBuffer buf)
{
    synchronized (readLock) {
        synchronized (stateLock) { … } … }
}
```

send & **receive** are similar

http://grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/8-b132/sun/nio/ch/DatagramChannelImpl.java#DatagramChannelImpl

Why would we care?

Firewall Traversal

SelectorImpl
&
Locks

Second verse, same as the first…

http://grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/8-b132/sun/nio/ch/SelectorImpl.java#SelectorImpl

FileChannel.transferTo & DatagramChannel

Platform Dependent, sure

`FileChannel.transferTo`

&

`DatagramChannel`

Observed on Mac:
call **`sendfile`**, get **`EINVAL`**, call **`sendto`**
over and over

SelectorImpl
&
HashSet Garbage

http://grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/8-b132/sun/nio/ch/SelectorImpl.java#SelectorImpl

```
public void handleFds(Selector selector)
{
    final Set<SelectionKey> keys =
        selector.selectedKeys();

    if (!keys.isEmpty())
    {
        final Iterator<SelectionKey> iter =
            keys.iterator();
        while (iter.hasNext()) { … }
    }
}
```

Whoa!… Java 8-ize!

```
public void handleFds(Selector selector)
{
    final Set<SelectionKey> keys =
        selector.selectedKeys();

    keys.forEach((key) -> { … });
}
```

HashSet usage means node garbage
so…
Borrow from netty.io & replace
`publicSelectedKeys` &
`selectedKeys`

https://github.com/netty/netty/blob/master/transport/src/main/java/io/netty/channel/nio/NioEventLoop.java
https://github.com/netty/netty/blob/master/transport/src/main/java/io/netty/channel/nio/SelectedSelectionKeySet.java


JAVA NIO
Y U NO LOVE LAMBDAS?
imgflip.com

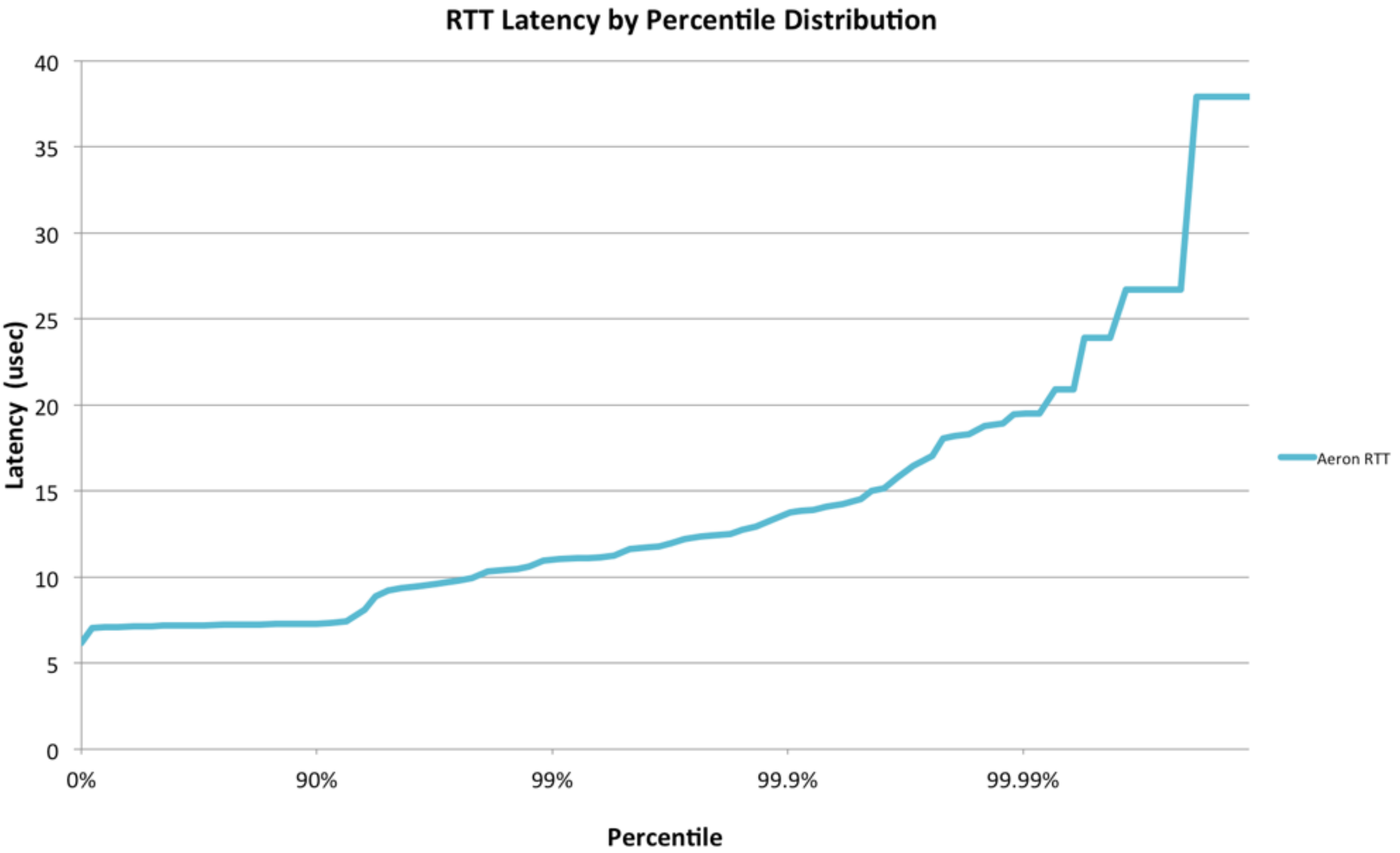
```
public void handleFds(Selector selector)
{
    // optimize underneath, more usable API
    selector.forEachSelectedKey(
        (key) ->
        {
            …
        });
}
```

What it could be…
and something optimized underneath

Just
the
tip
...
but
some
times
...

But what can happen when you get it right?

20+ million msgs/sec
single stream (NO contention)
40 byte messages


RTT Latency by Percentile Distribution
Latency (usec)
40
35
30
25
20
15
10
5
0
0%
90%
99%
99.9%
99.99%
Percentile
Aeron RTT

Not Just Java

C++, Erlang, Go, …

InfiniBand, PCI-e 3.0, …

Questions?

- Aeron *https://github.com/real-logic/Aeron*
- Kaazing *http://www.kaazing.com*
- SlideShare *http://www.slideshare.com/toddleemontgomery*
- Twitter @toddlmontgomery

Thank You!