Using Modern C++ In Anger

Todd L. Montgomery
@toddlmontgomery

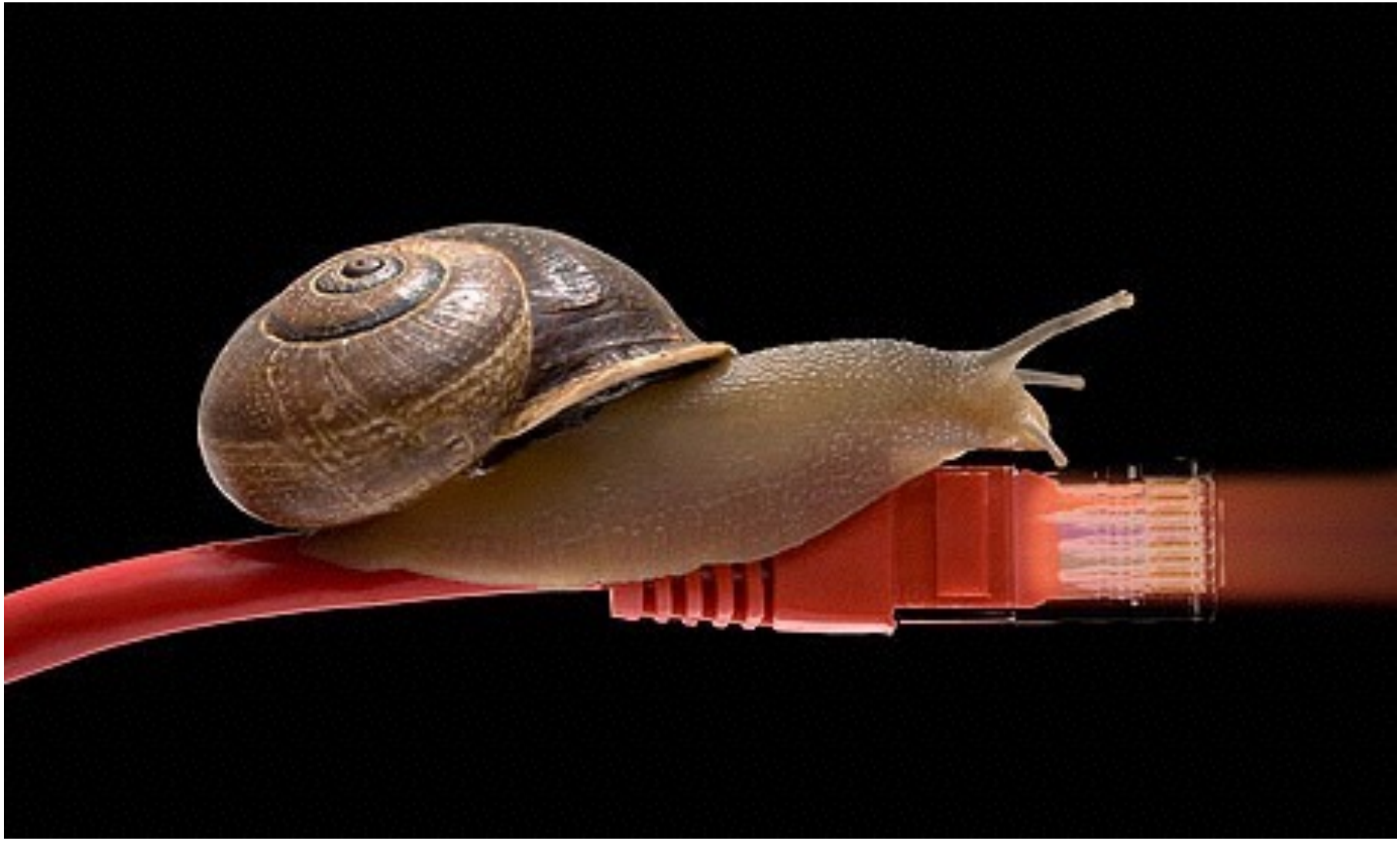
- Aeron and C++?
- What constitutes Modern C++?
- How Aeron Adopts Modern C++?
- What Lessons were learned?
- What's Next?

Aeron and C++?

Truly modern messaging transport

Feature Bloat & Complexity

Not Fast Enough



Low & Predictable Latency is key

We are in a new world

Multi-core, Multi-socket, Cloud...

We are in a new world

Multi-core, Multi-socket, Cloud...

UDP, IPC, InfiniBand, RDMA, PCI-e

Aeron is trying a new approach

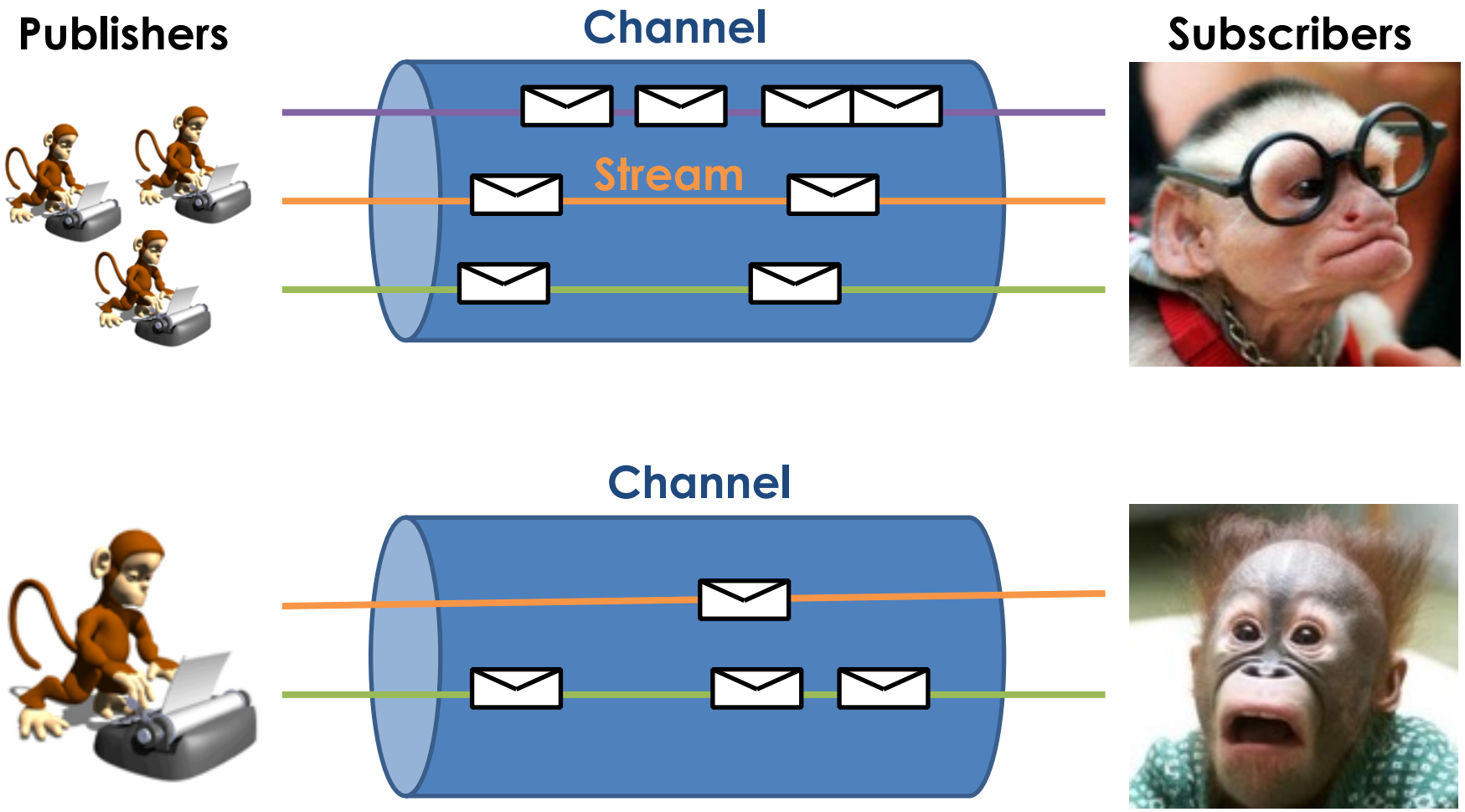
The Team

Todd Montgomery

Richard Warburton

Martin Thompson

Messaging



A library, not a framework, on which other abstractions and applications can be built

Composable Design

OSI layer 4 Transport for message oriented streams

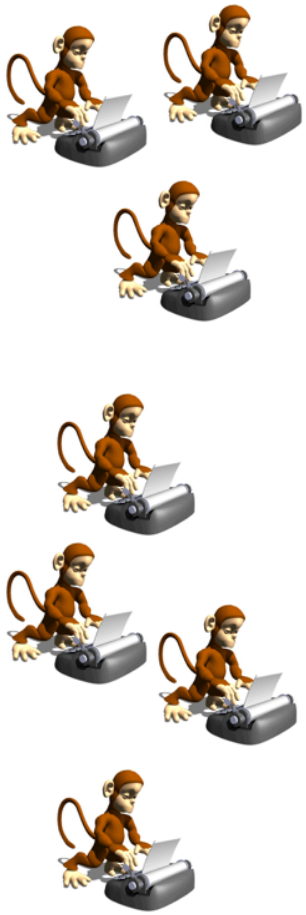
OSI Layer 4 (Transport) Services

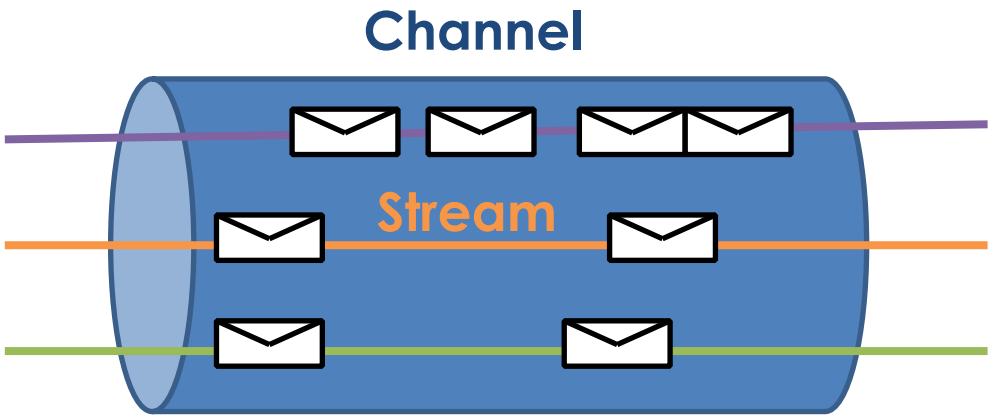
1. Connection Oriented Communication
2. Reliability
3. Flow Control
4. Congestion Avoidance/Control
5. Multiplexing

Multi-Everything World!

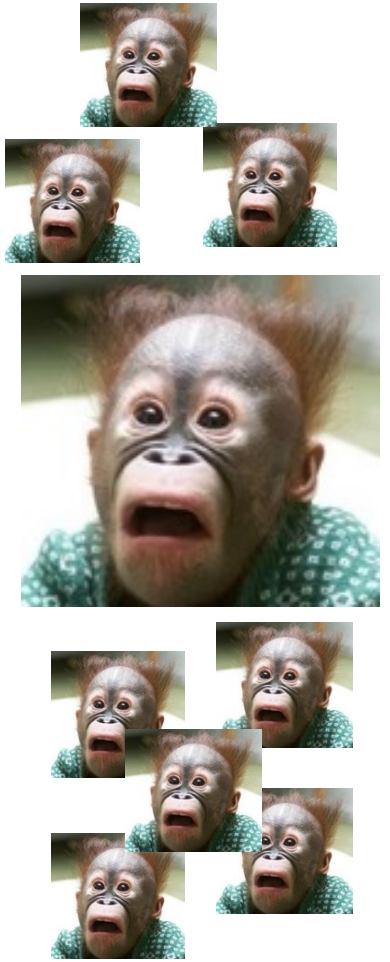
Multi-Everything World

Publishers





Subscribers

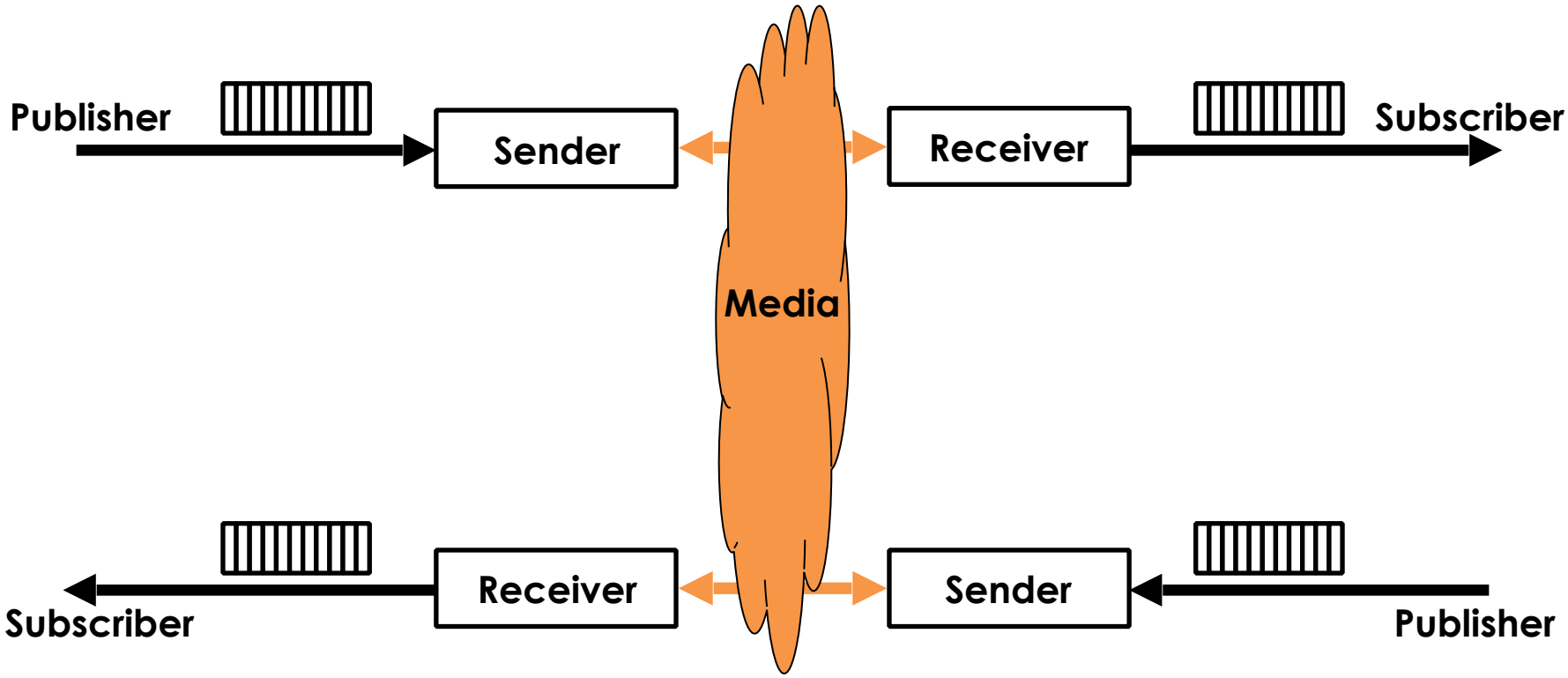


Design Principles

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

Architecture

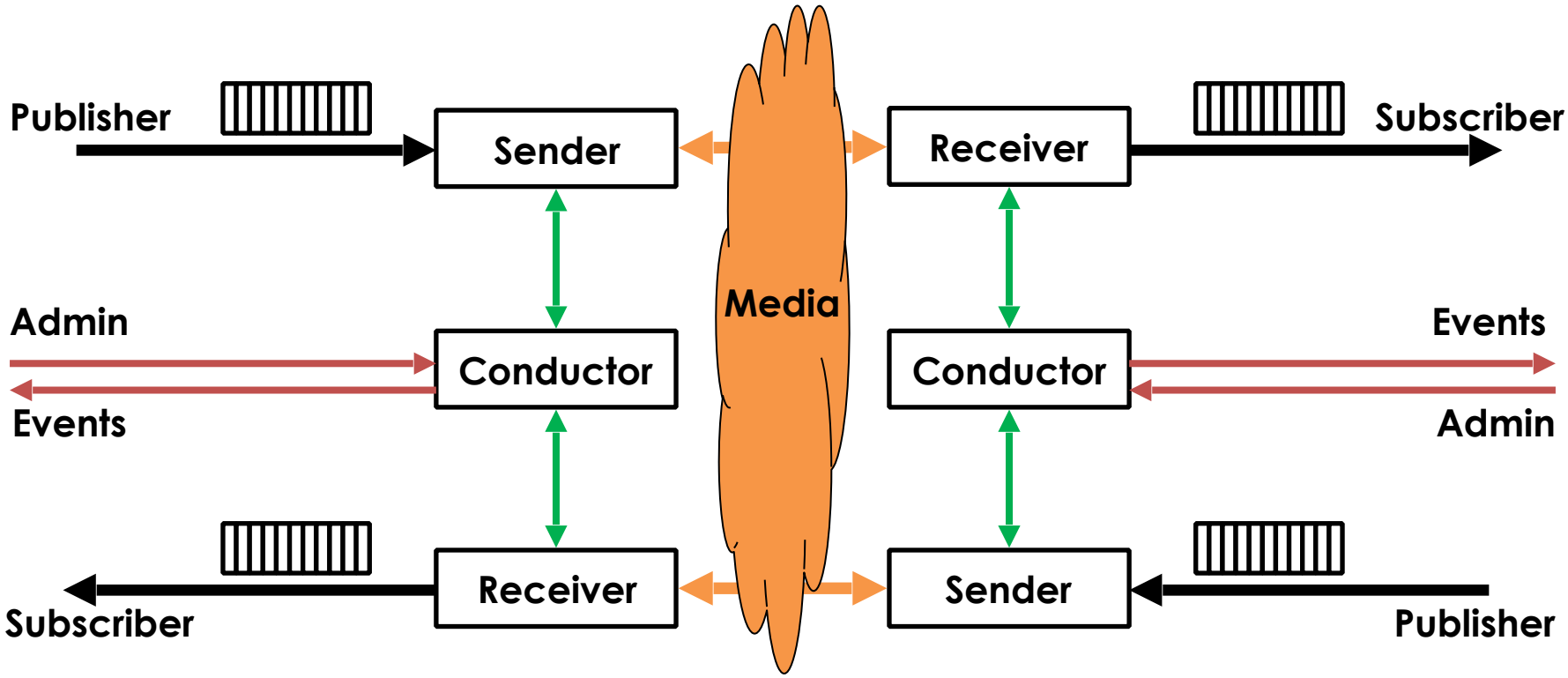
Architecture



IPC Log Buffer

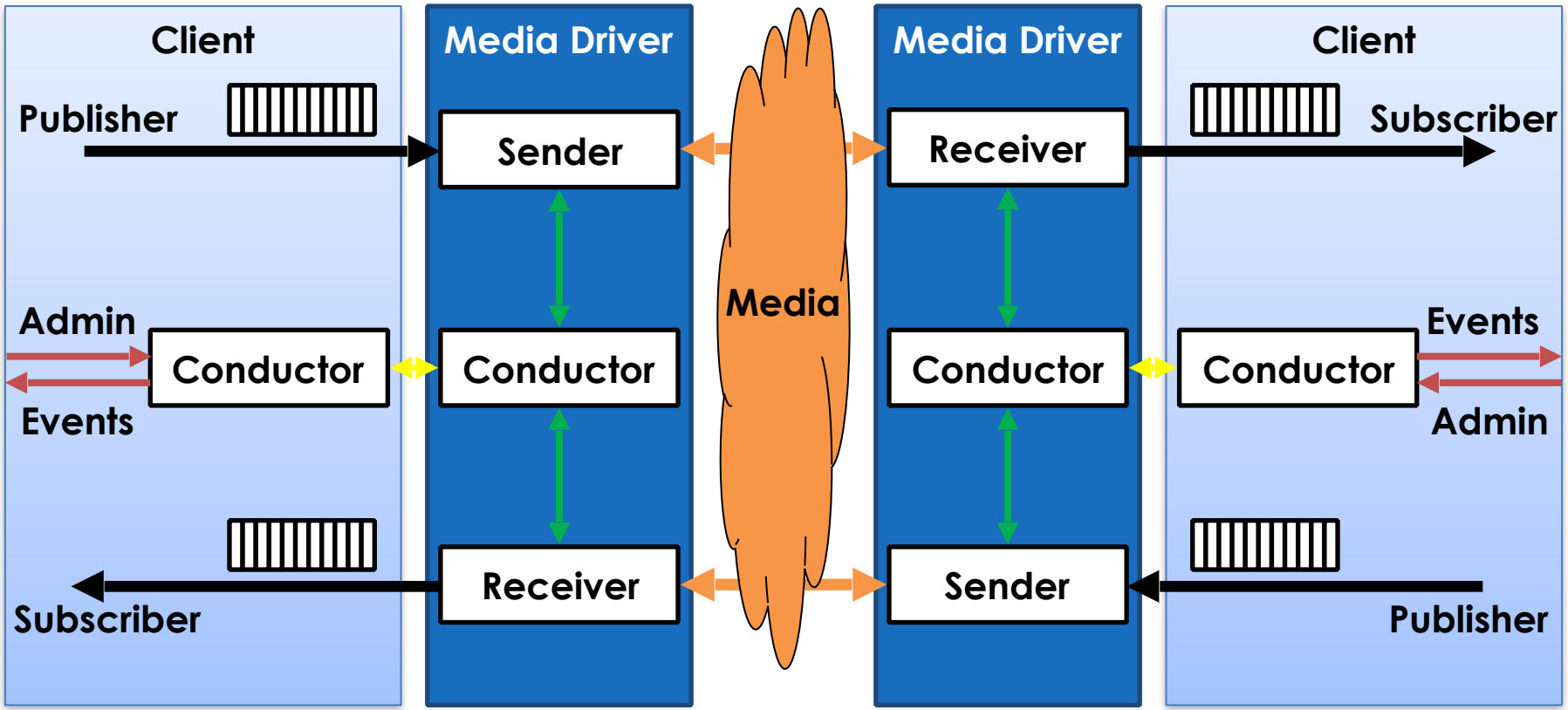
Media (UDP, InfiniBand, PCI-e 3.0)

Architecture



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

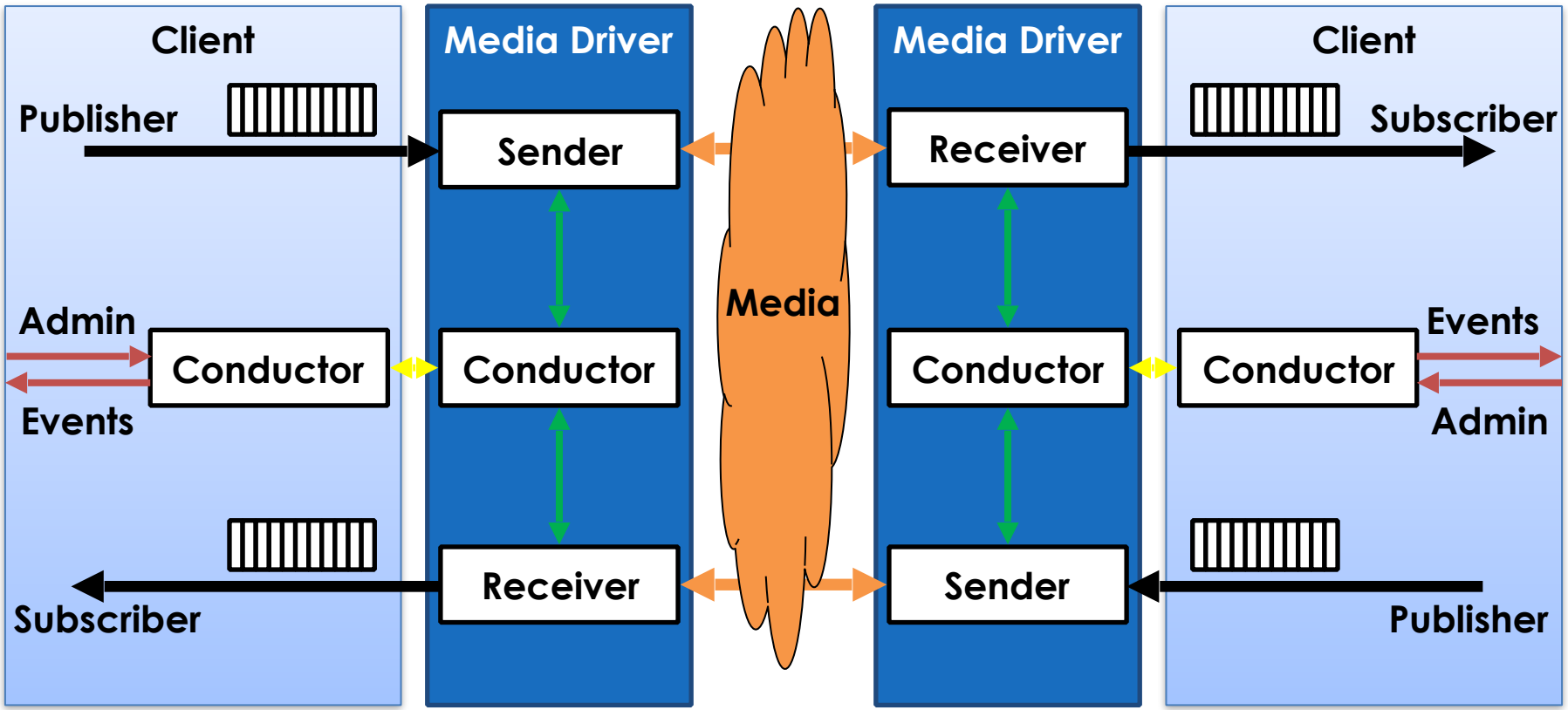
Architecture



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

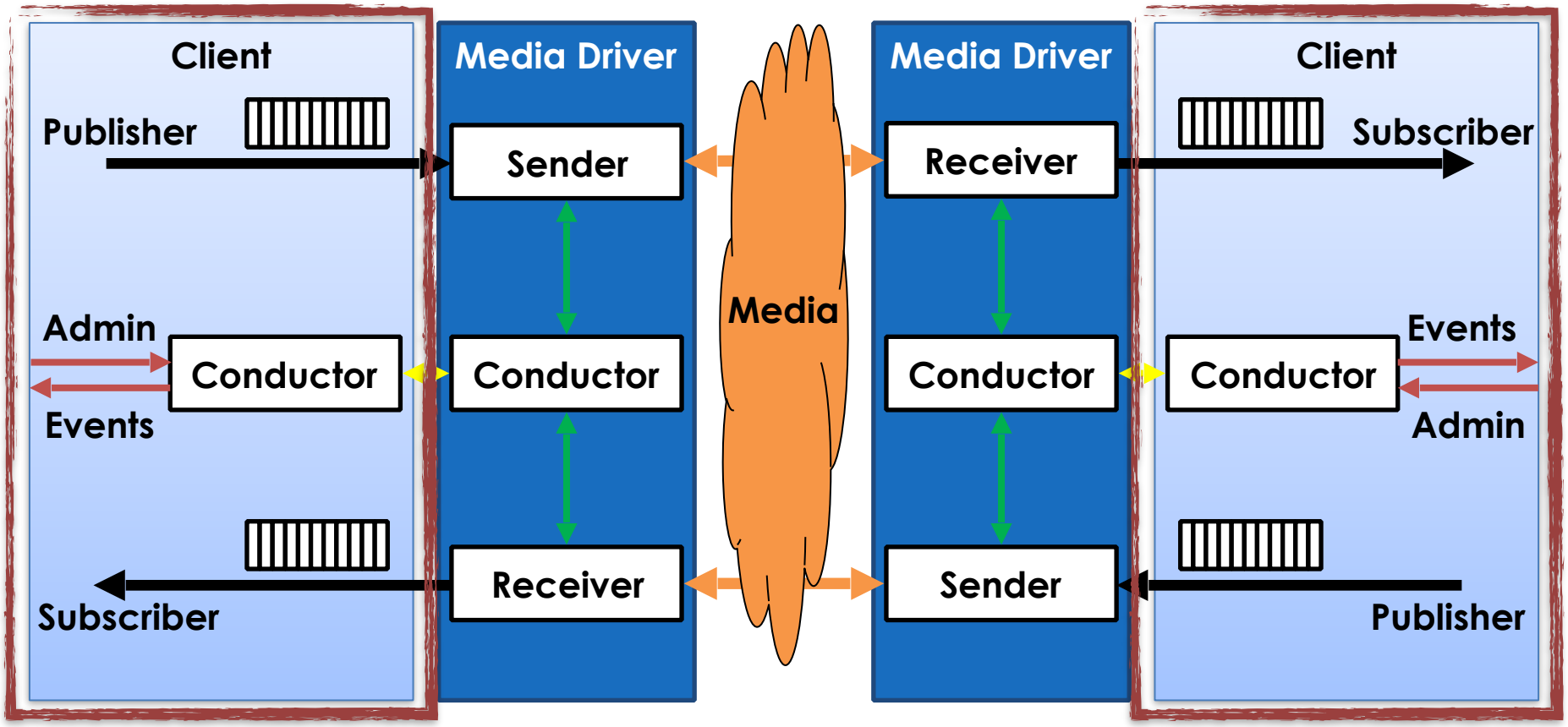
C++?

Architecture



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

Architecture



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

Client

- Application API
- Java, C, C++, C#, etc.

The Media Driver is (currently) Java…


DON'T
PANIC!

What constitutes Modern C++?

O'REILLY®

Effective Modern C++

42 SPECIFIC WAYS TO IMPROVE YOUR USE OF C++11 AND C++14

Scott Meyers

Modern C++

Resource Ownership & Lifetime

Modern C++

- Idioms (RAII, etc.)

Resource Acquisition is Initialization (RAII)

```
std::lock_guard
std::unique_ptr
```

http://en.cppreference.com/w/cpp/language/raii

Modern C++

- Idioms (RAII, etc.)
- Smart Pointers

Smart Pointers

```
std::shared_ptr
std::unique_ptr
 std::weak_ptr
```

http://en.cppreference.com/w/cpp/memory

Modern C++

- Idioms (RAII, etc.)
- Smart Pointers
- Lambda's & Function Objects

Modern C++

- Idioms (RAII, etc.)
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)

Atomic Operations

```
std::atomic<bool>
std::atomic_flag
```

http://en.cppreference.com/w/cpp/atomic

Modern C++

- Idioms (RAII, etc.)
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)
- Thread Support

Thread Support

```
std::thread
std::mutex
std::promise
std::future
```

http://en.cppreference.com/w/cpp/thread

Modern C++

- Idioms (RAII, etc.)
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)
- Thread Support
- Move Construction/Assignment

Modern C++

- Idioms (RAII, etc.)
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)
- Thread Support
- Move Construction/Assignment
- Toolchain (Build/Test)

Modern Toolchain

CMake
Google Test
Google Mock
etc.

https://github.com/google/googletest
https://cmake.org/

How Aeron Adopts Modern C++?

Smart Pointers

```
std::shared_ptr<Aeron> aeron = Aeron::connect(context);

std::int64_t id = aeron->addPublication(channel, streamId);

std::shared_ptr<Publication> publication = aeron->findPublication(id);
```

Lambda's & Function Objects

```
context.newSubscriptionHandler(
    [](const std::string& channel, std::int32_t streamId, std::int64_t correlationId)
    {
        // ...
    });
```

Thread Support

```
std::thread rateReporterThread([&](){ rateReporter.run(); });
```

```
{
    std::lock_guard<std::recursive_mutex> lock(m_adminLock);
    // ..
}
```

What Lessons were learned?

Lessons - What do you think?

- Idioms (RAII, etc.)
- Stack Allocation
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)
- Thread Support
- Move Construction/Assignment
- Toolchain (Build/Test)

Lessons

- Idioms (RAII, etc.)
- Stack Allocation
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)
- Thread Support
- Move Construction/Assignment
- Toolchain (Build/Test)


TAMGDEYA.R

RAII & Smart Pointers

- Give in to Smart Pointers

RAII & Smart Pointers

- Give in to Smart Pointers
- Explicit Coupling

RAII & Smart Pointers

- Give in to Smart Pointers
- Explicit Coupling
- Explicit Scoping

Stack Allocation (Lack of)

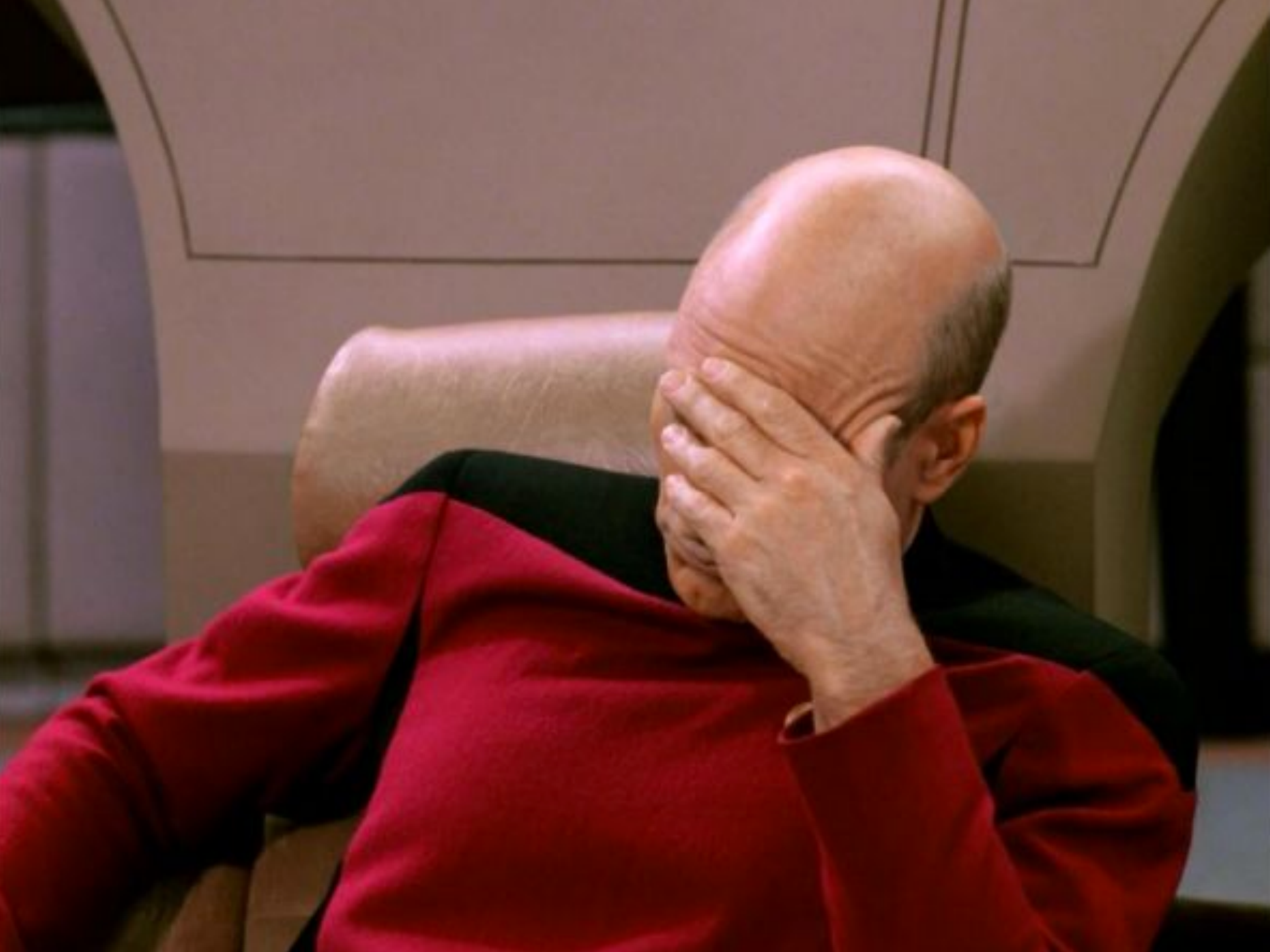
[insert excuse Y]

…

Escape Analysis

Value Types

…



DOUBLE FACEPALM

FOR WHEN ONE FACEPALM DOESN'T CUT IT

Lessons

- Idioms (RAII, etc.)
- Stack Allocation
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)
- Thread Support
- Move Construction/Assignment
- Toolchain (Build/Test)


No, sir.
I do not like to "move it, move it."

Move Construction/Assignment

- **Much more than you think**

Move Construction/Assignment

- Much more than you think
- Sometimes/Often better to copy

Move Construction/Assignment

- Much more than you think
- Sometimes/Often better to copy
- Optimization Interactions

Move Construction/Assignment

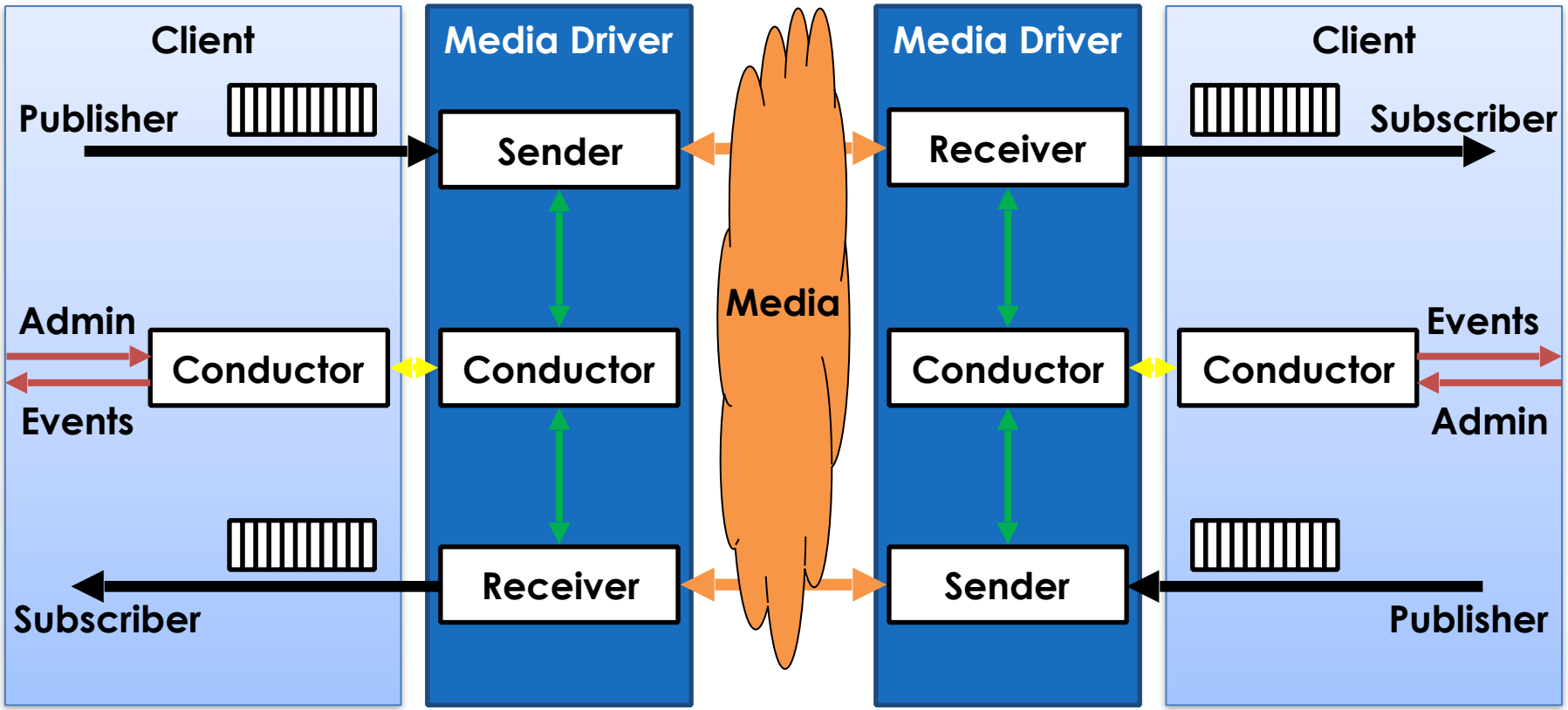
- Much more than you think
- Sometimes/Often better to copy
- Optimization Interactions
- When you have to do it… why?!?

Lessons

- Idioms (RAII, etc.)
- Stack Allocation
- Smart Pointers
- Lambda's & Function Objects
- Atomics (`std::atomic`)
- Thread Support
- Move Construction/Assignment
- Toolchain (Build/Test)


ATOMICS!

Architecture



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

Data Structures (Shared Memory)

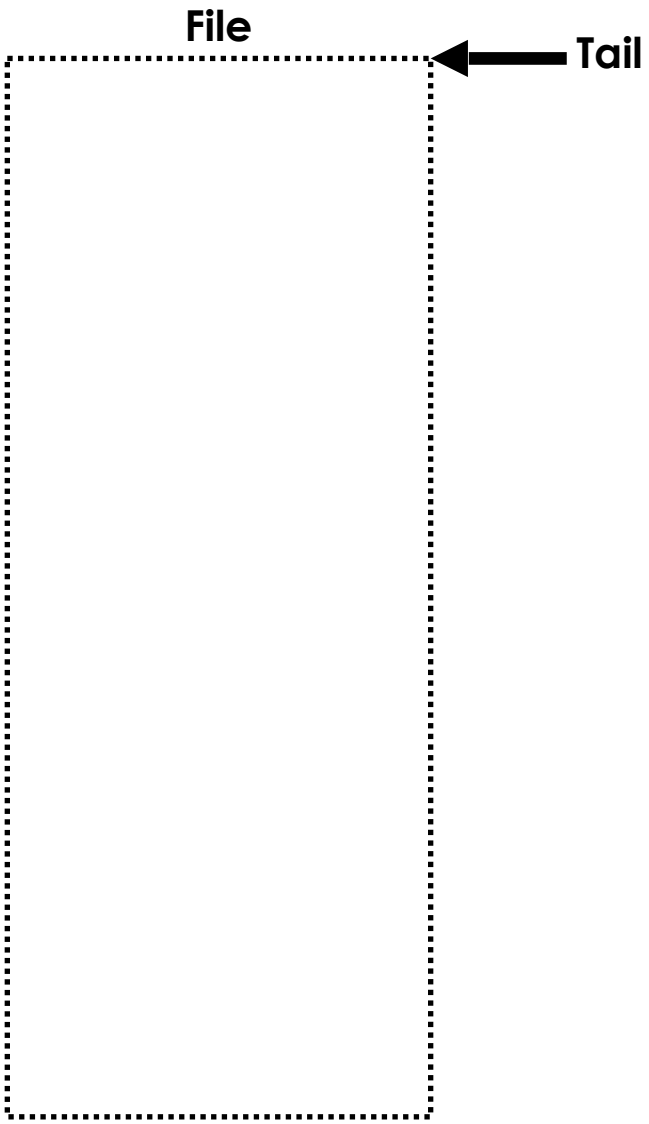
- IPC Ring Buffers
- IPC Broadcast Buffers
- IPC Log Buffers

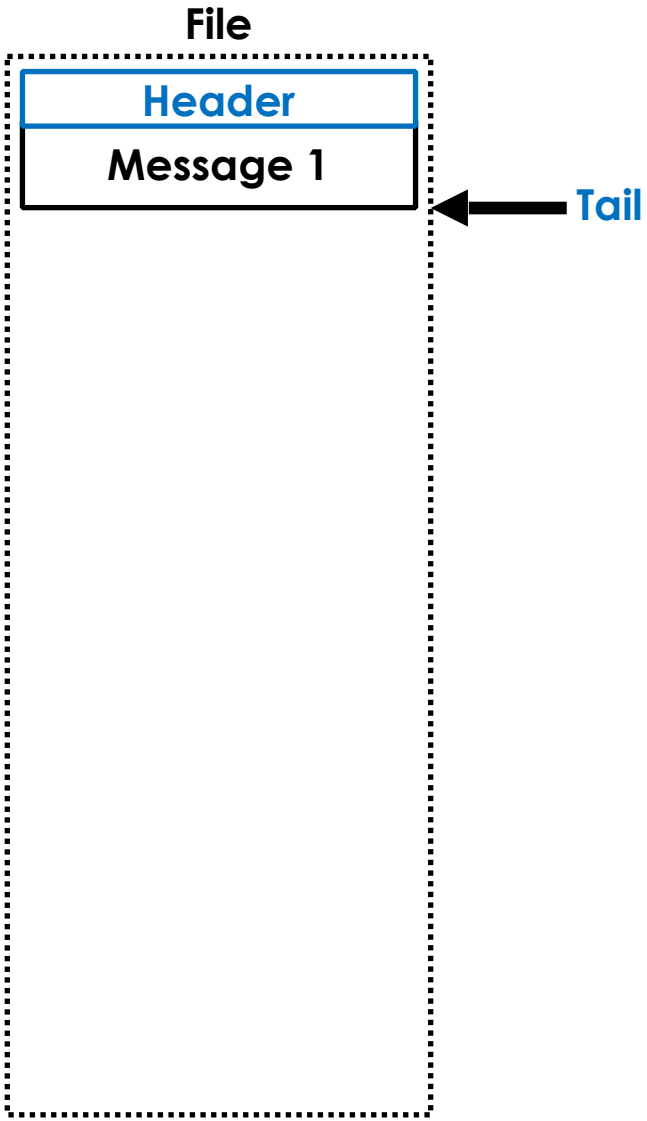
What Aeron does

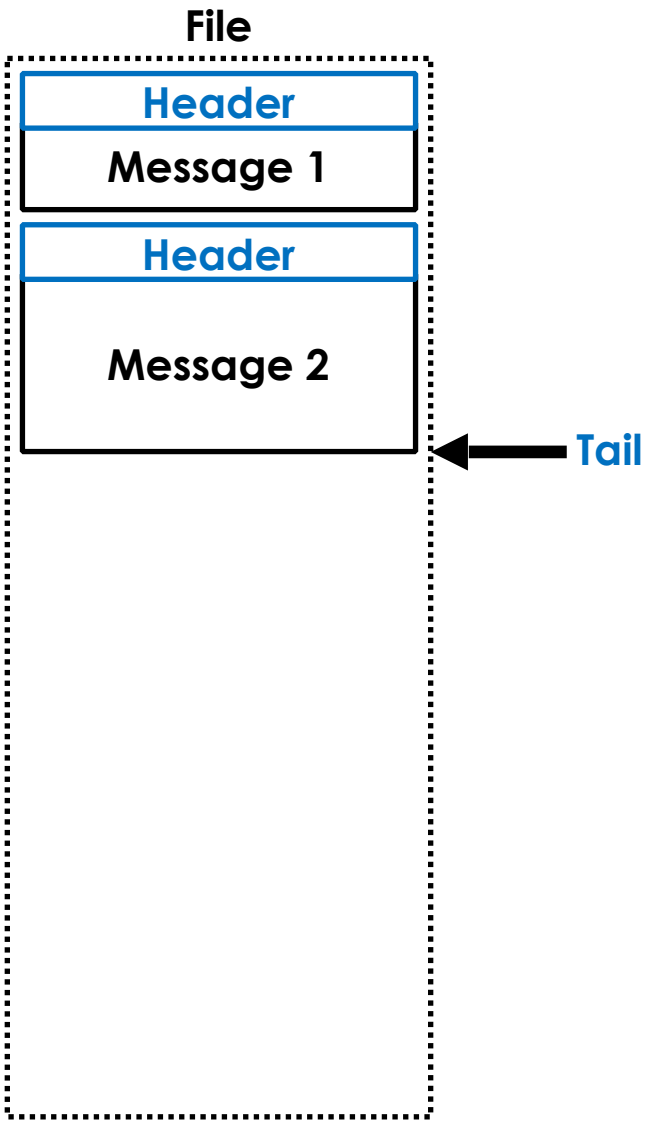
Creates a

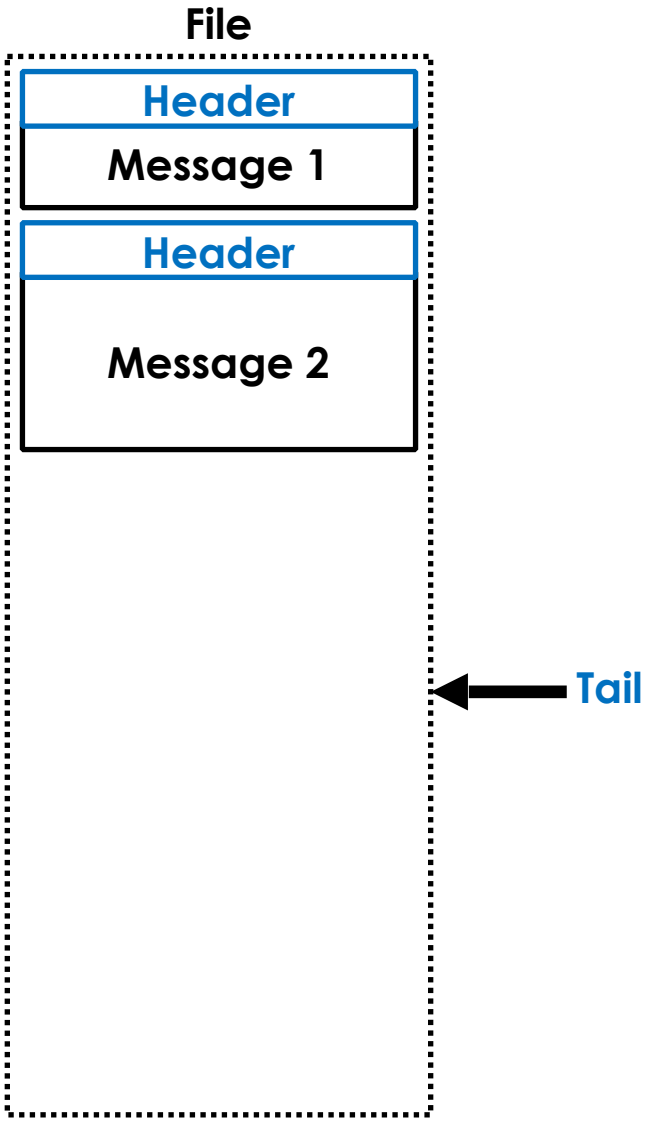
replicated persistent log

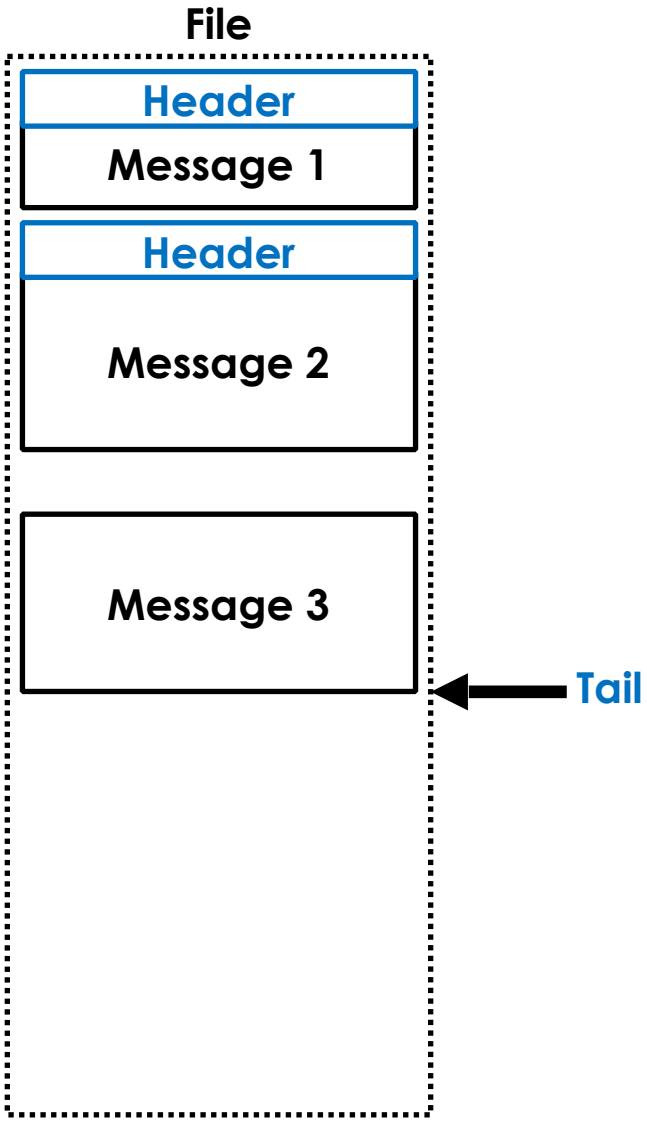
of messages

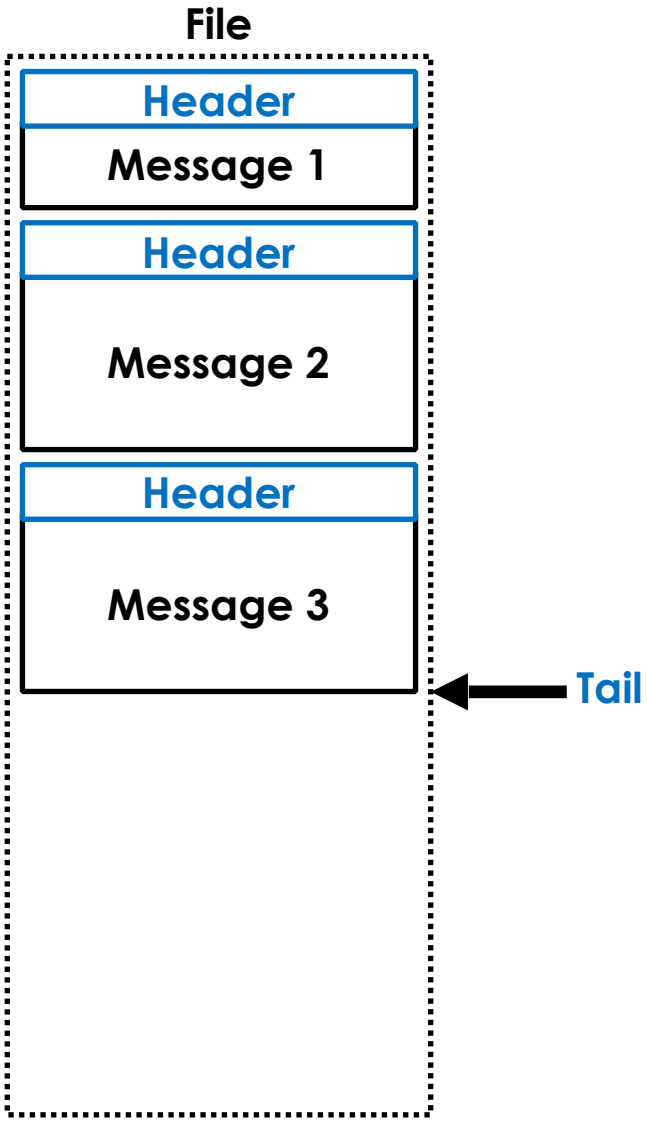

File
Tail

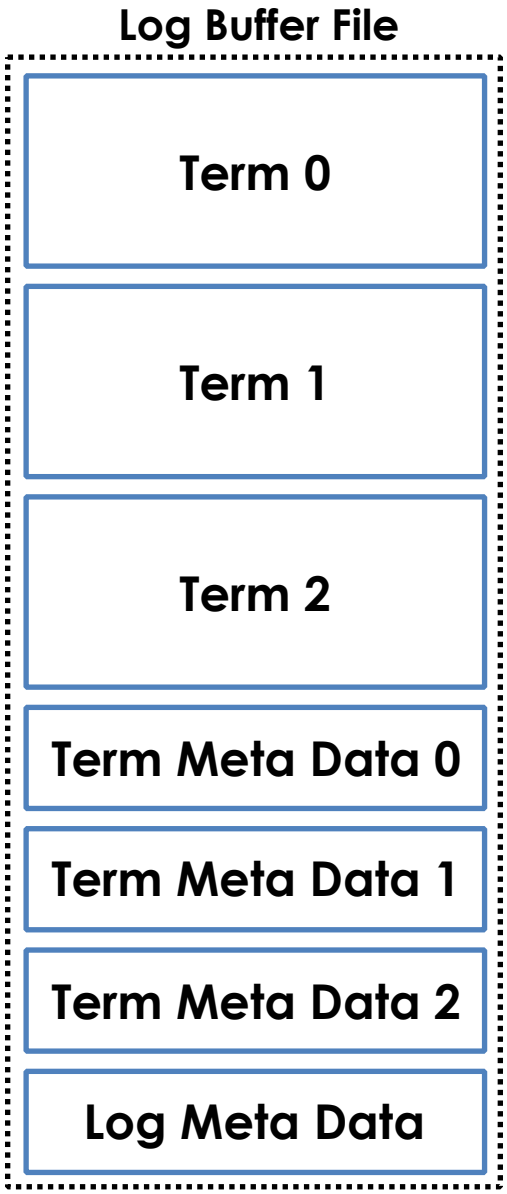

File
Header
Message 1
Tail

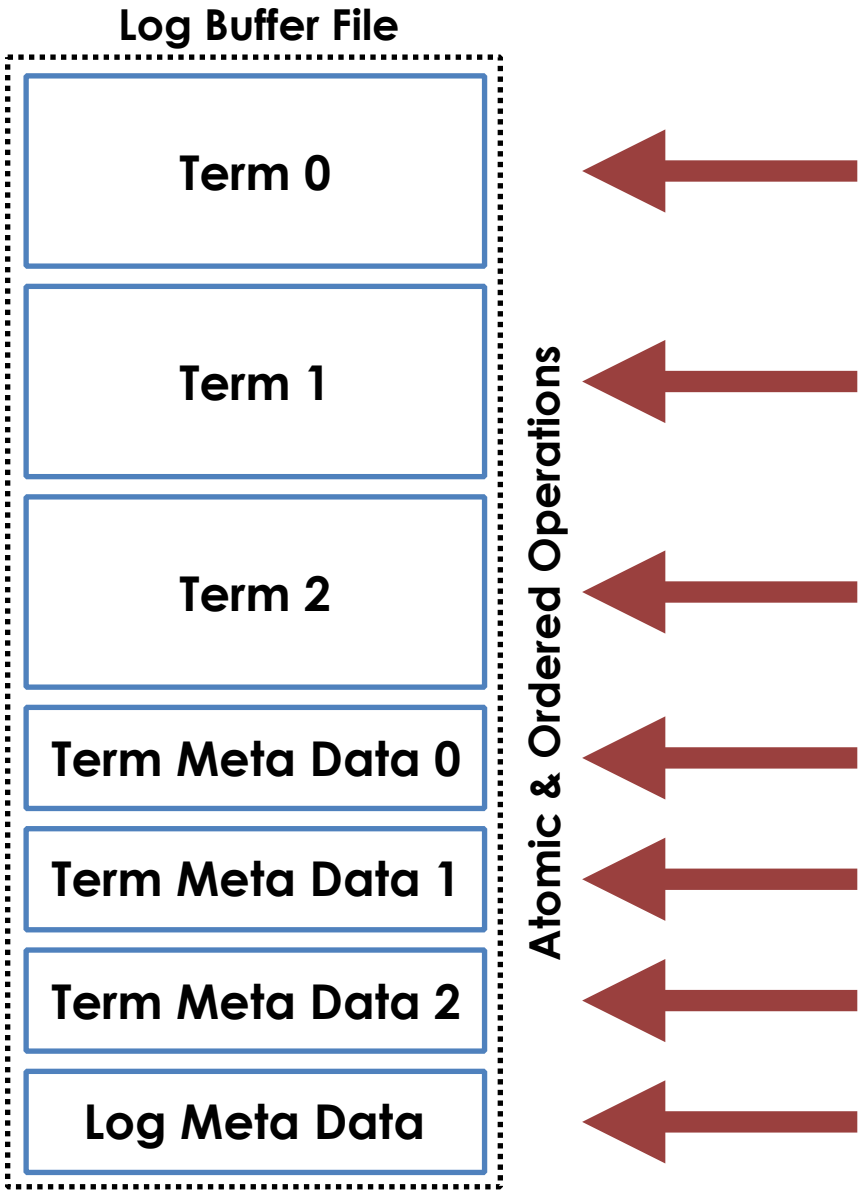

File
Header
Message 1
Header
Message 2
Tail


File
Header
Message 1
Header
Message 2
Tail


File
Header
Message 1
Header
Message 2
Message 3
Tail


File
Header
Message 1
Header
Message 2
Header
Message 3
Tail


Log Buffer File
Term 0
Term 1
Term 2
Term Meta Data 0
Term Meta Data 1
Term Meta Data 2
Log Meta Data


Log Buffer File
Term 0
Term 1
Term 2
Term Meta Data 0
Term Meta Data 1
Term Meta Data 2
Log Meta Data
Atomic & Ordered Operations

Position

Unique identification of a byte within each stream

Publishers, Senders, Receivers, and Subscribers all keep position counters

Position counters are the key to flow control and monitoring

Statistics & Position Counters are accessible in shared memory

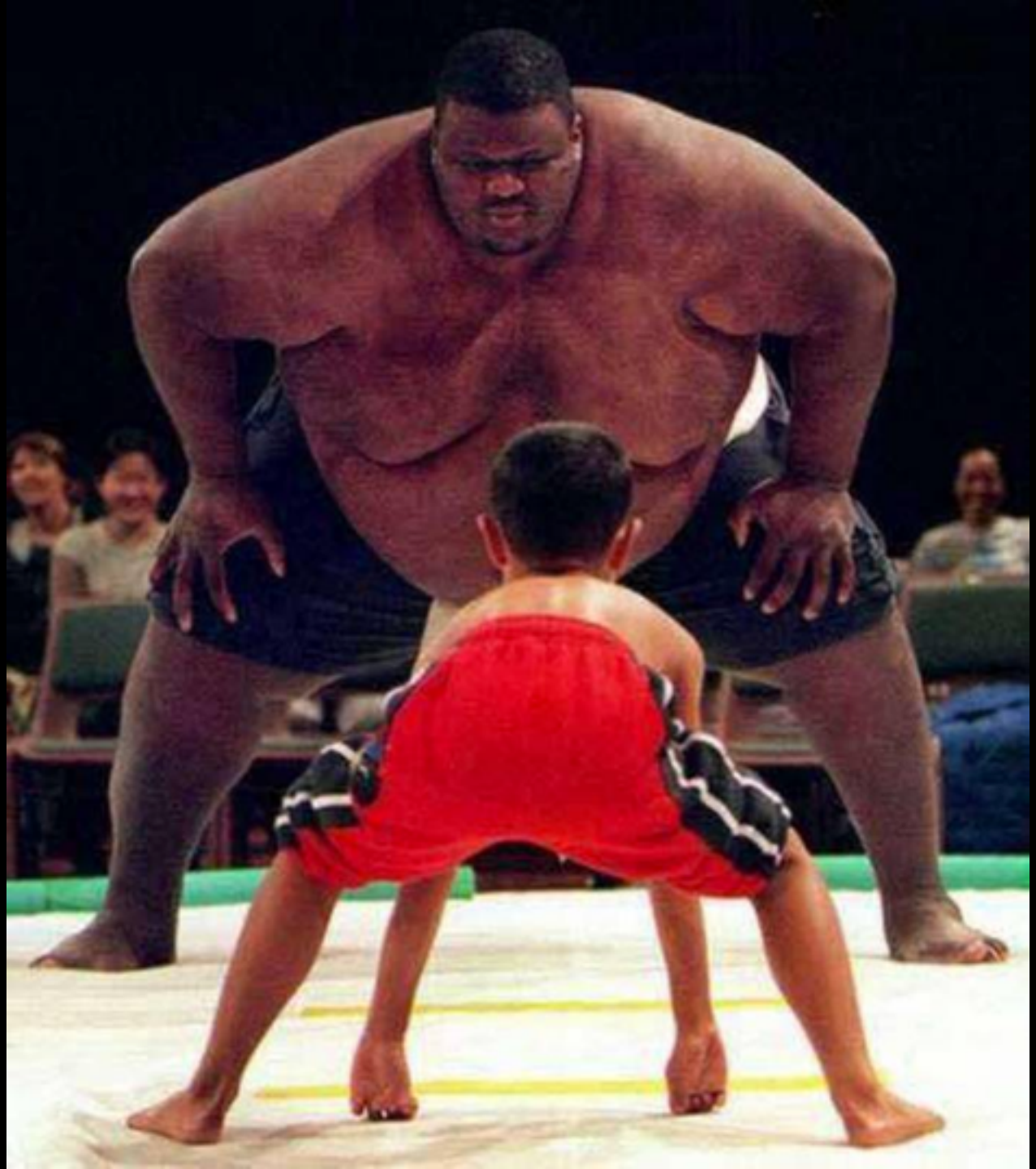
Atomic & Ordered Operations

Multiple Challenges

- Size (and Layout)
- Memory Models (C++11 to Java)

Size

Matters



Size Matters (Atomics)

- Not designed for arbitrary memory
- Size not the same as the types
- Concerned only with operations

Memory Models

- Interoperability with JMM
- `std::memory_order` fit for purpose

So…

Aeron uses its own atomic operations C/C++ functions (JMM compatible)

* gcc x86_64 initially contributed by phaynes (phaynes@threatmetrix.com)

What's Next?

Finished a few passes of Profiling and Tuning

IPC, 32-byte Messages

C++ to C++

32+ Million messages per second

IPC, 32-byte Messages

C++: 32M msg/sec

Java: 30M msg/sec

.NET: 15M msg/sec

Persistence

Replication

Efficient FEC

Encryption/Security

Services

Aeron Core

1.0!!

Performance

C/C++ Driver

Multi Unicast Send

In closing…


Do epic shit,
or die trying.

Where can I find it?

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

Questions?

Aeron: *https://github.com/real-logic/Aeron*
Twitter: **@toddlmontgomery**

Thank You!