Design Draft

Efflux

Keeping the data flowing

Brian Ritchie

https://github.com/dotnetpowered

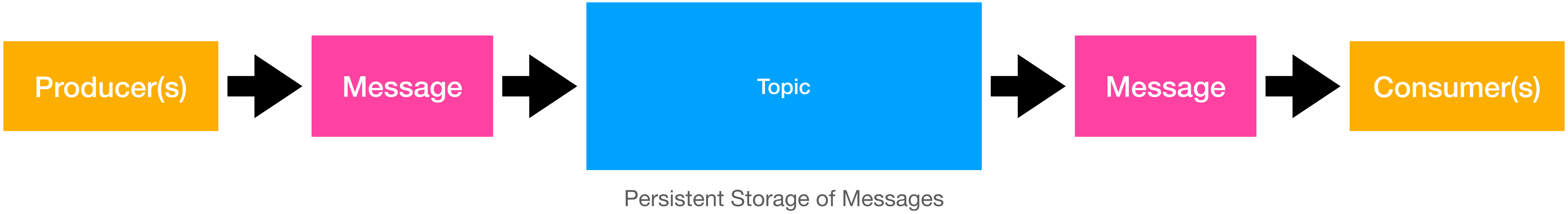
Efflux

Highlight Reel

- Immutable data stream of messages
- Stores opaque data blocks
- “Block Chain” - Hashed contents linked to hashed contents of prior block
- Stores searchable metadata

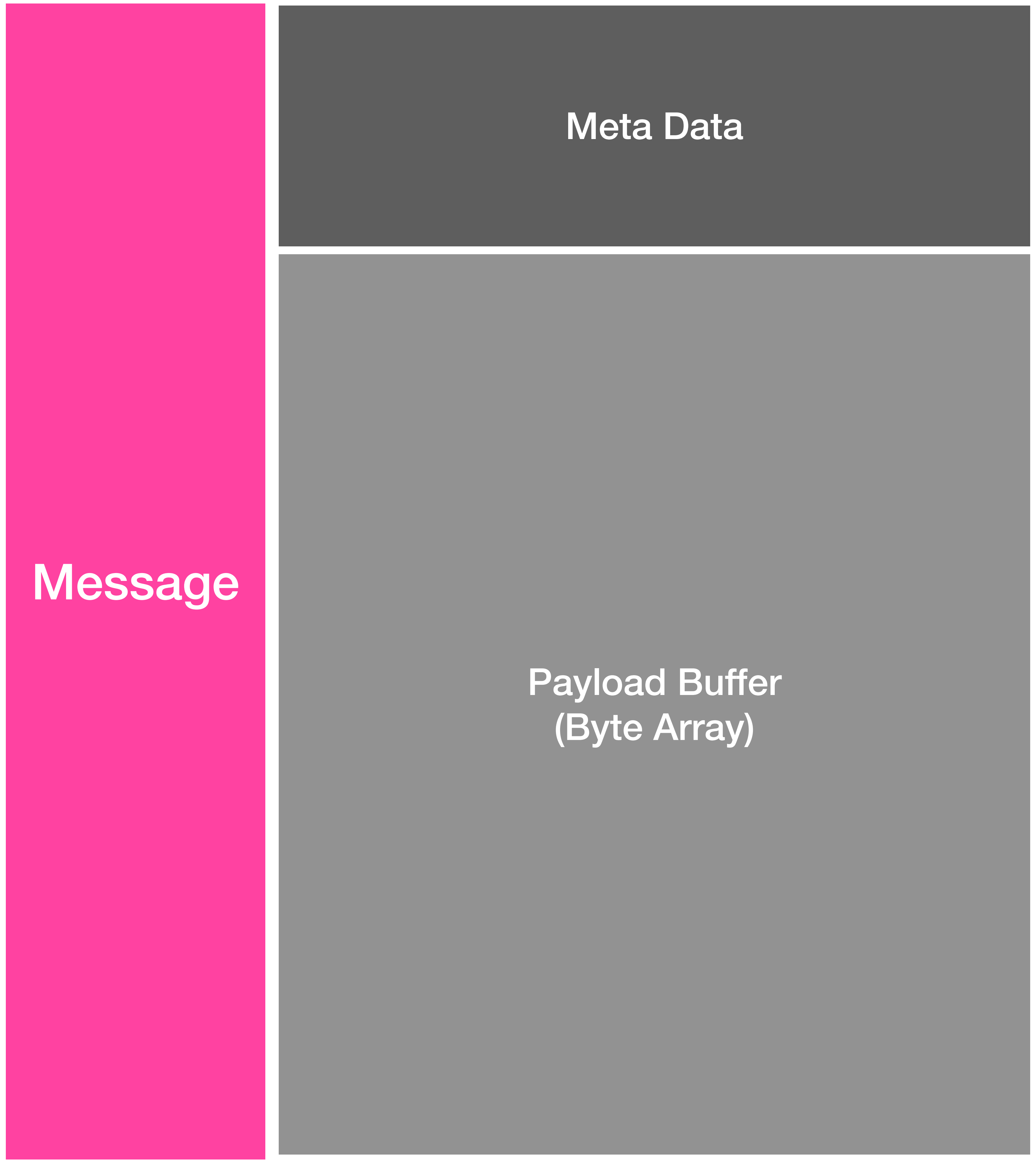
Efflux Particle

The flow of a particle



Efflux Particle

What's in a particle?

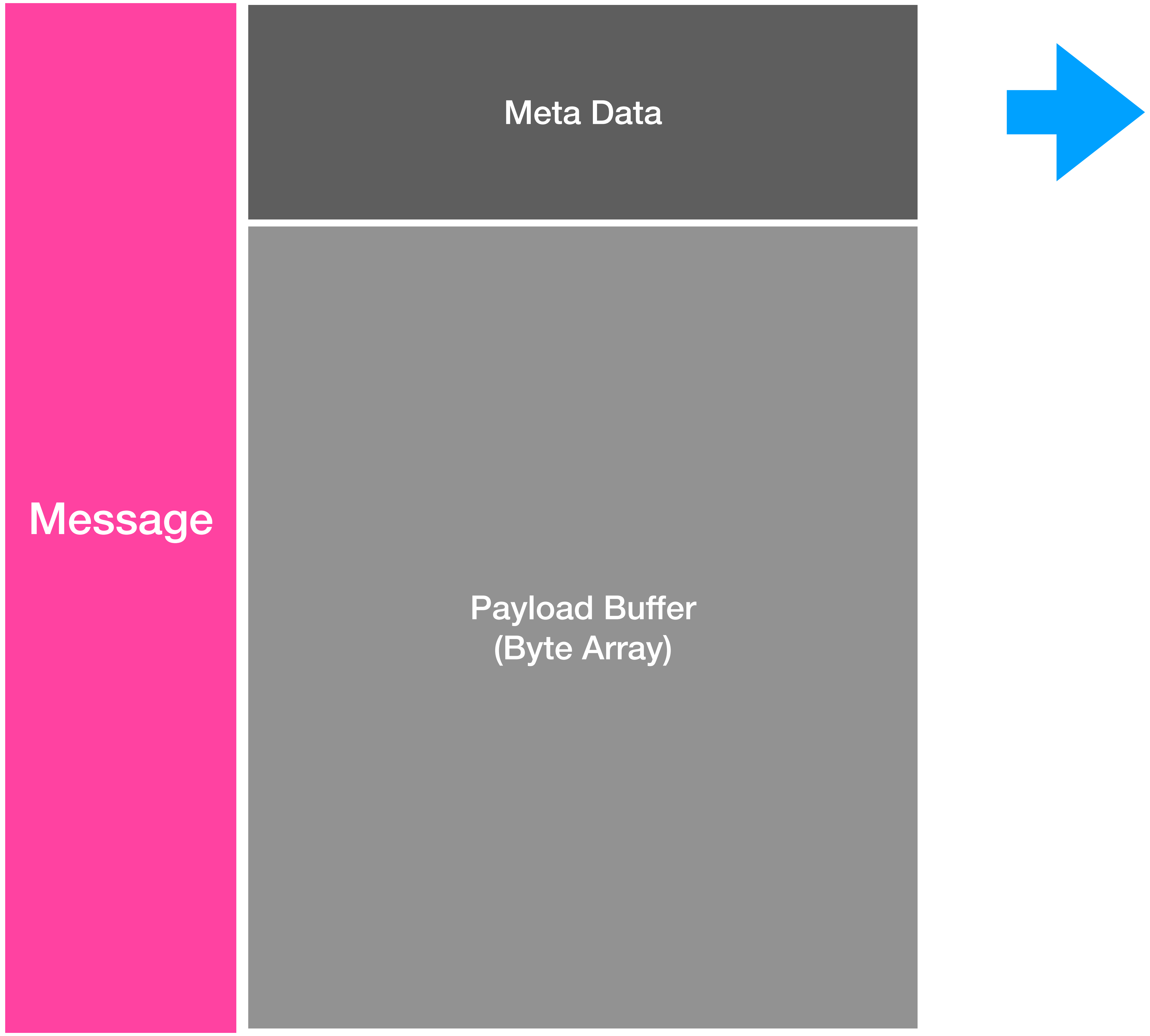


Structured data about the message.

Opaque data block storing message.

Efflux Particle

Meta Data



Message

Meta Data

Payload Buffer
(Byte Array)

Meta Data Contents

Common Data

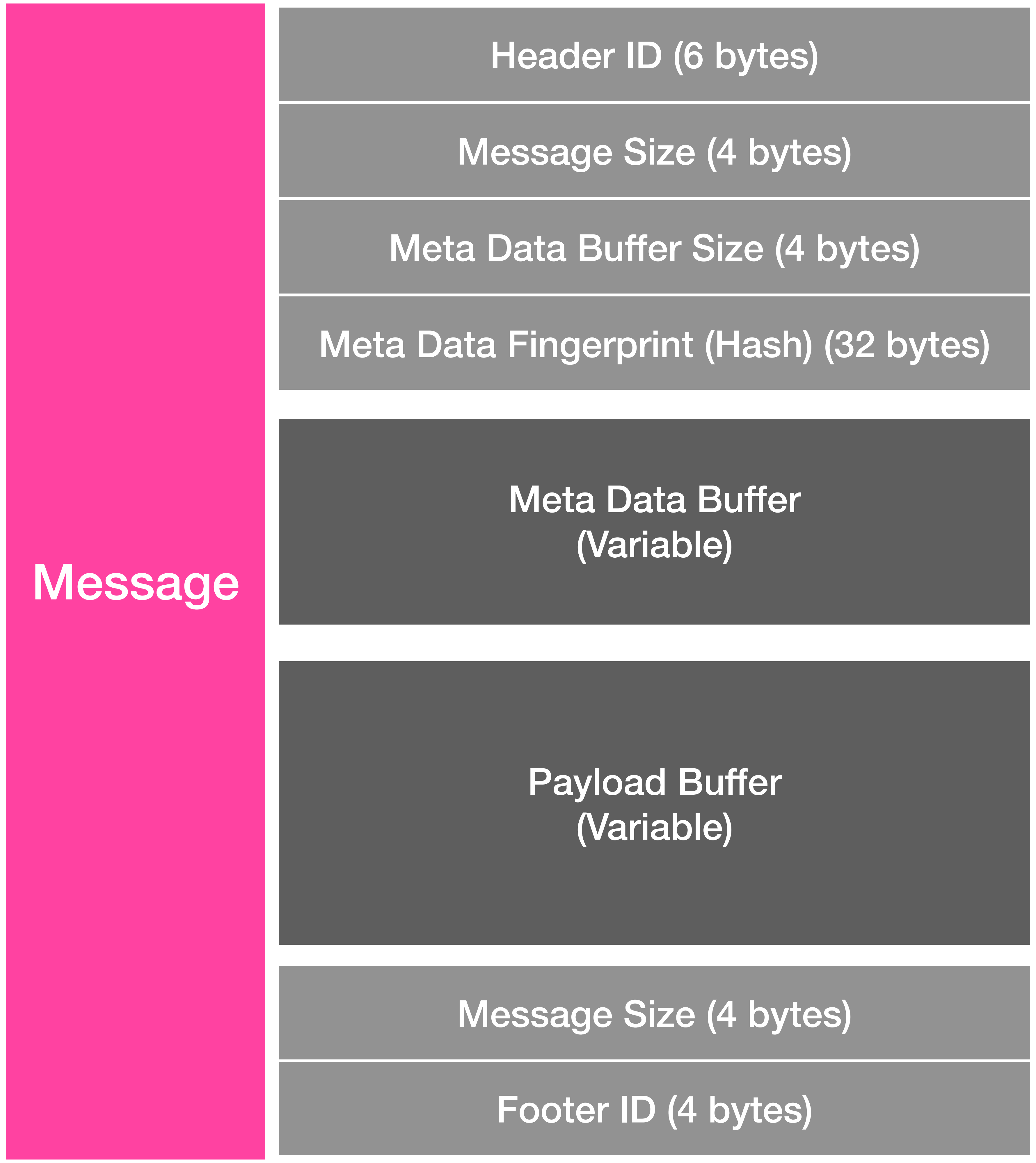
```
// Unique Message ID (GUID/UUID)
string Id

// UTC DateTime when Message was created
DateTime Created

// UTC DateTime when Data was created
public DateTime Timestamp

// SHA256 Hash of data
public byte[] PayloadHash

// Fingerprint of previous record (SHA256 hash)
byte[] LinkedFingerprint

// Application-specific data type
string DataType

// application/json, application/octet-stream, text/plain;UTF8
string ContentType

// gzip
string ContentEncoding

string MessageGroup
```

Custom Data

```
IDictionary<string, string> Properties
```

Efflux Particle

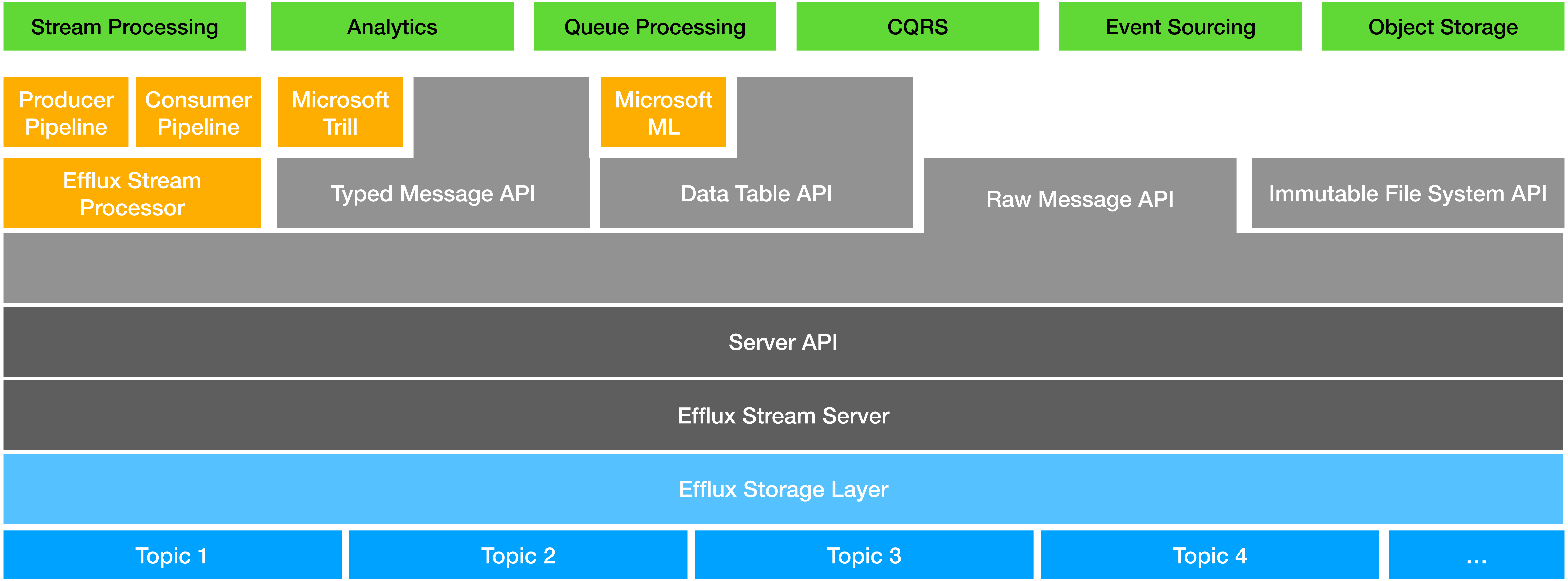
Binary Storage Format



Message	Header ID (6 bytes)	
	Message Size (4 bytes)	
	Meta Data Buffer Size (4 bytes)	
	Meta Data Fingerprint (Hash) (32 bytes)	
	Meta Data Buffer (Variable)	Serialized as JSON
	Payload Buffer (Variable)	Raw data (bytes) - see ContentType for data representation
	Message Size (4 bytes)	
	Footer ID (4 bytes)	

Efflux Layered Architecture

One platform, many use cases



Stream Processing

Analytics

Queue Processing

CQRS

Event Sourcing

Object Storage

Producer Pipeline

Consumer Pipeline

Microsoft Trill

Microsoft ML

Efflux Stream Processor

Typed Message API

Data Table API

Raw Message API

Immutable File System API

Server API

Efflux Stream Server

Efflux Storage Layer

Topic 1

Topic 2

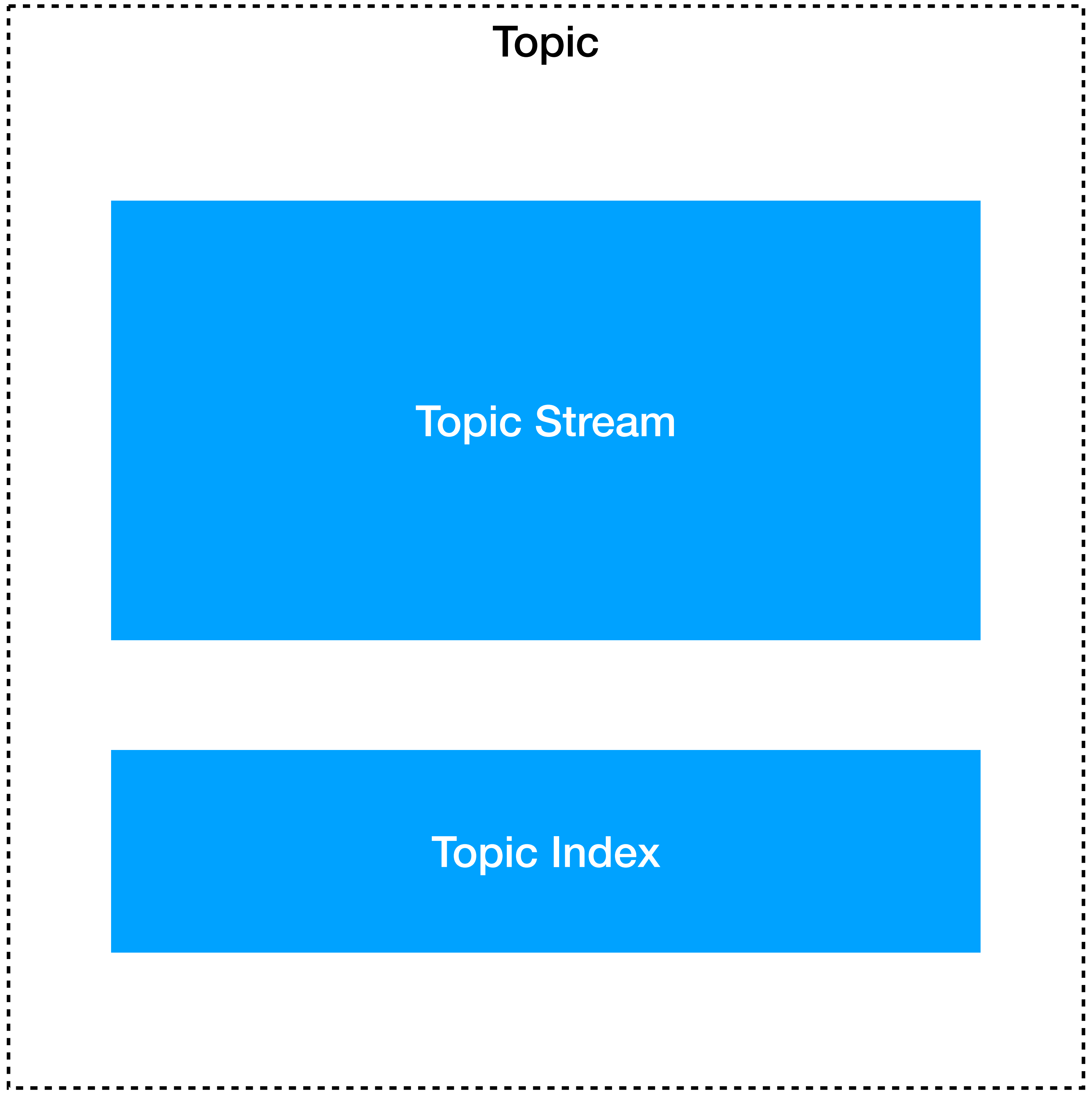
Topic 3

Topic 4

...

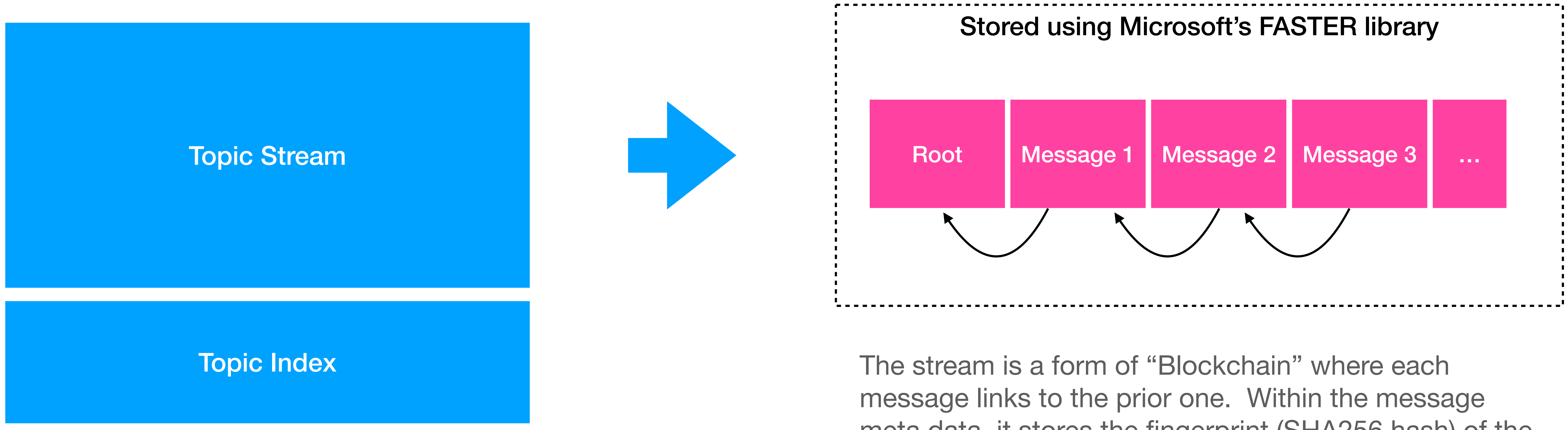
Efflux Topic

Storage



Efflux Topic

Storage - Topic Stream



Topic Stream

Topic Index

Stored using Microsoft’s FASTER library

Root

Message 1

Message 2

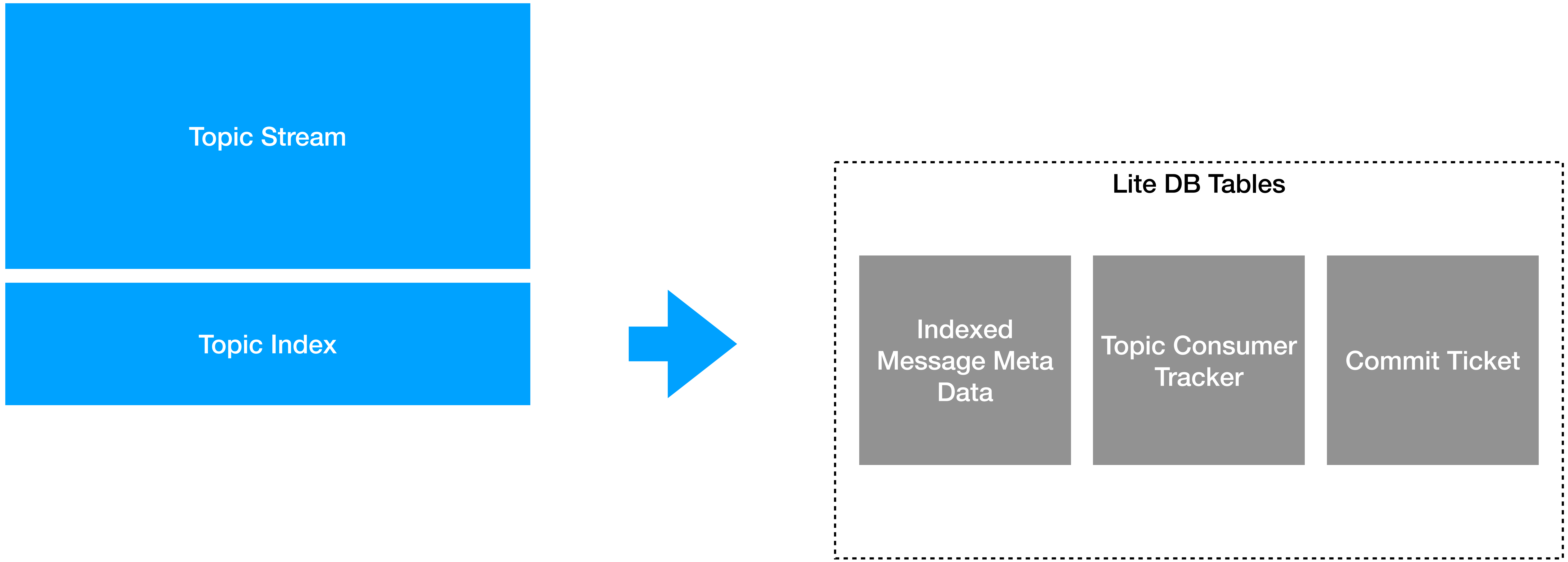
Message 3

...

The stream is a form of “Blockchain” where each message links to the prior one. Within the message meta data, it stores the fingerprint (SHA256 hash) of the prior message. This becomes part of the content that is hashed to produce the fingerprint of the message.

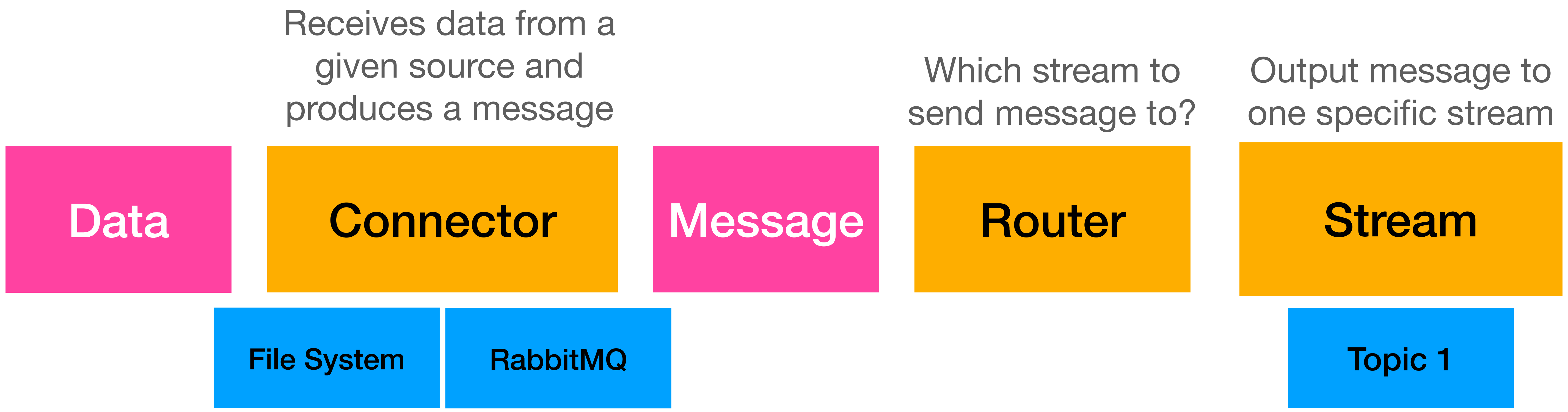
Efflux Topic

Storage - Topic Index



Efflux Stream Processor

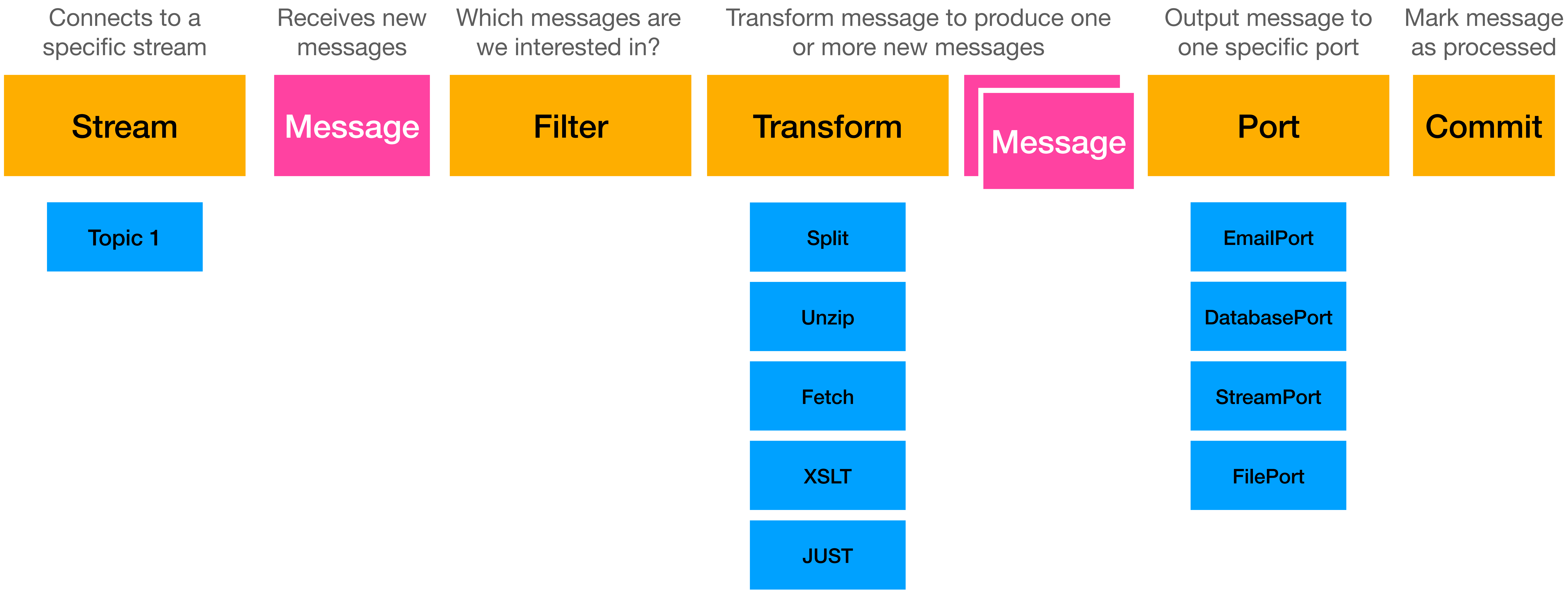
Producer Pipeline



Receives data from a given source and produces a message
Which stream to send message to?
Output message to one specific stream
Data
Connector
Message
Router
Stream
File System
RabbitMQ
Topic 1

Efflux Stream Processor

Consumer Pipeline



Connects to a specific stream

Stream

Topic 1

Receives new messages

Message

Which messages are we interested in?

Filter

Transform message to produce one or more new messages

Transform

Split

Unzip

Fetch

XSLT

JUST

Message

Output message to one specific port

Port

EmailPort

DatabasePort

StreamPort

FilePort

Mark message as processed

Commit