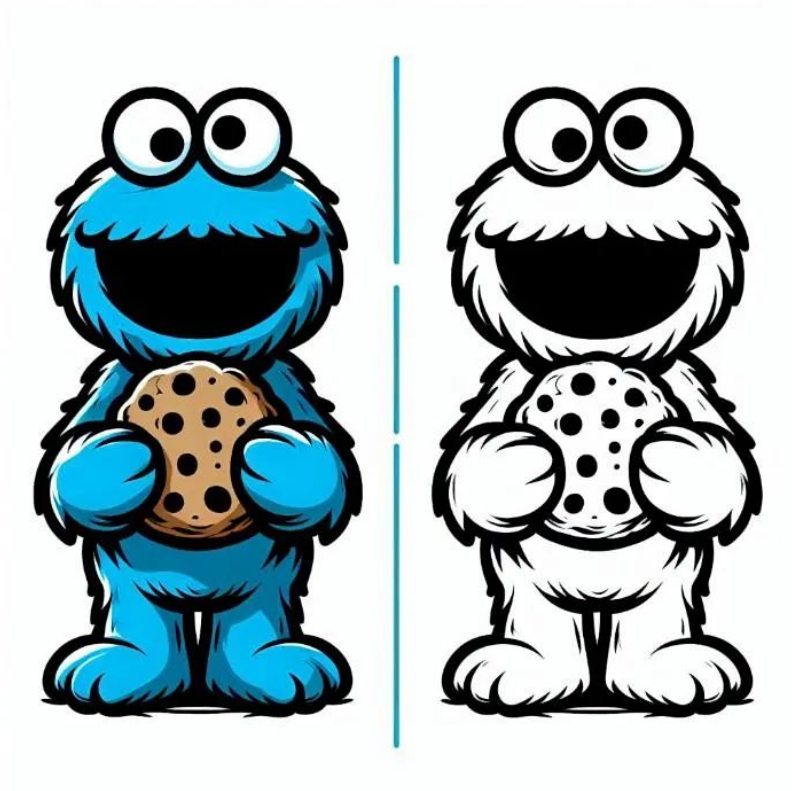
Unlocking the Power of Communication: The Intriguing World of Message Brokers

Part 1 of N…

The concepts of

Message Brokers / Event Brokers

The concepts of Message Brokers / Event Brokers



The concepts of Message Brokers / Event Brokers

What is a Message Broker / Event Broker?

The concepts of Message Brokers / Event Brokers

What is a Message Broker / Event Broker?

What the differences between Brokers?

The concepts of Message Brokers / Event Brokers

What is a Message Broker / Event Broker?

What the differences between Brokers?

Why we use Message Brokers?

The concepts of Message Brokers / Event Brokers

What is a Message Broker / Event Broker?

What the differences between Brokers?

Why we use Message Brokers?

How to start?

The concepts of Message Brokers / Event Brokers

What is a Message Broker / Event Broker?

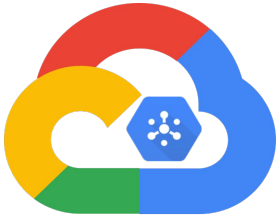
What the differences between Brokers?

Why we use Message Brokers?

How to start?

What’s next? Enjoy!

Popular Message / Event Brokers



What is a Message Broker?

Using RabbitMQ as an example:

RabbitMQ (AMQP) Model

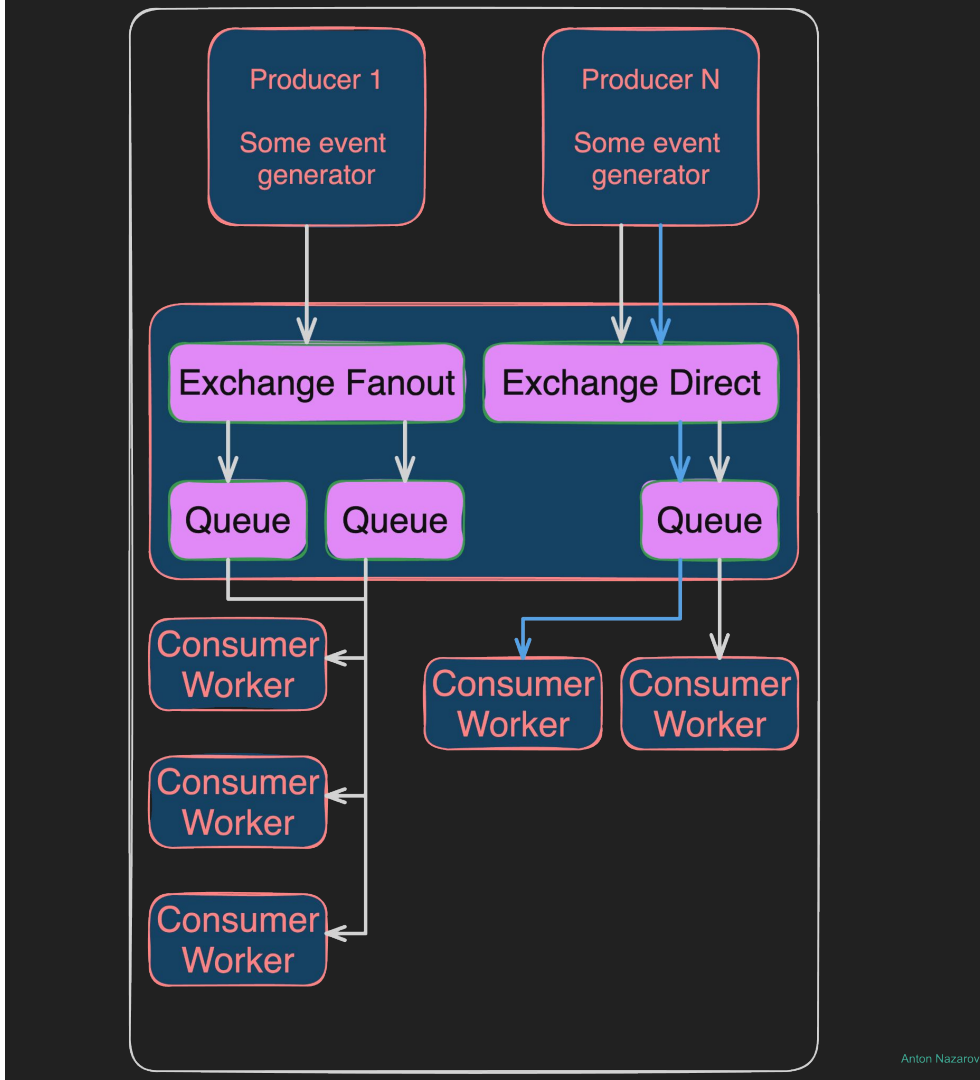
- **Exchange** with Fanout, Direct, and Topic types provide different strategies for message distribution.
- **Queue** stores messages until they are consumed by consumers.
- **Binding** specifies how messages are routed from an exchange to a queue.

Additional concepts:

- Produce/Consumer
- Channel
- Connection
- Virtual host
-

Use Cases:

- Microservices Communication
- Log Aggregation, Processing, Monitoring
- Event Driven Architectures
- Task Queues and Work Distribution
- High Availability and Disaster Recovery



What is a Event Broker?

Using Kafka as an example:

Kafka Model

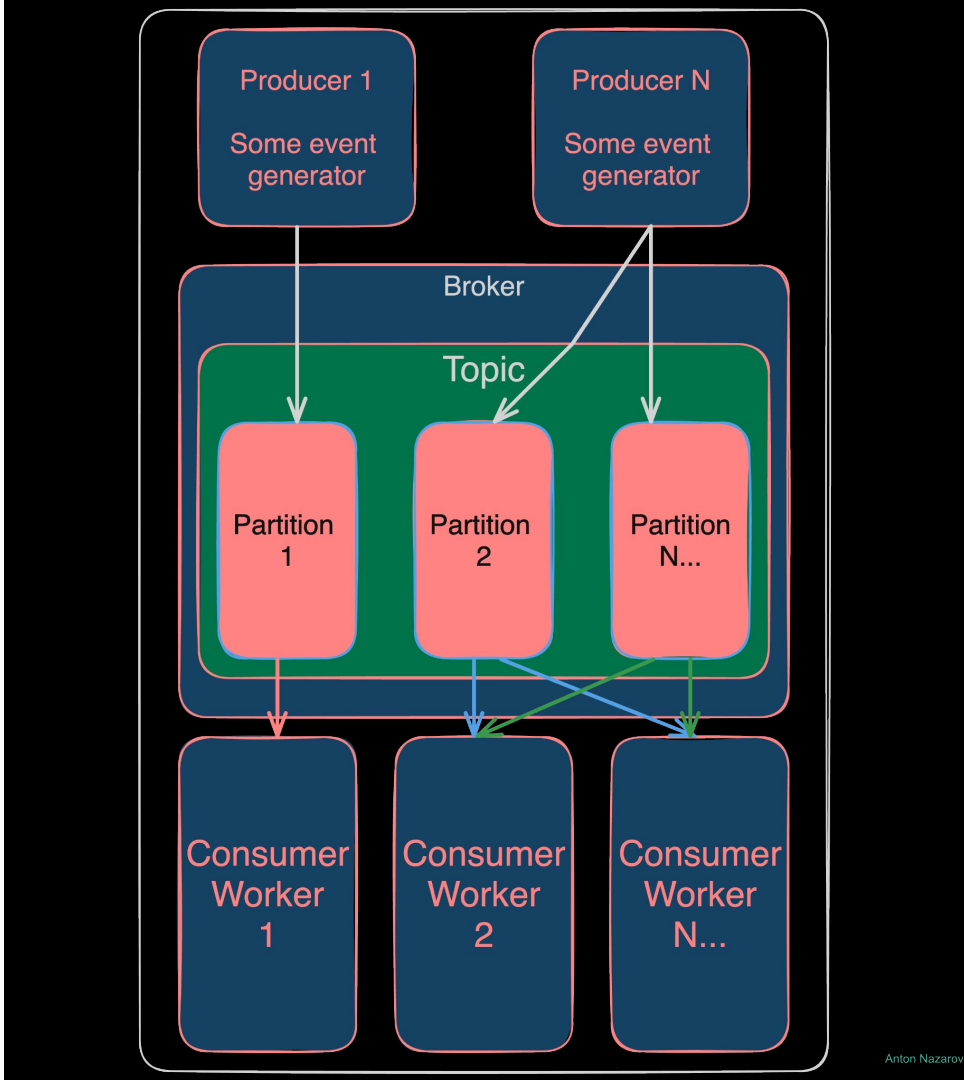
- **Topic** - represents a logical channel and message container for organising and partitioning data streams.
- **Partition** - are ordered, immutable sequence of messages.
- **Broker** - server responsible for storing and managing partitions.

Additional concepts:

- Produce/Consumer/Consumer Groups
- Offset
- Retention and Compaction
- Streams/Connect APIs

Use Cases:

- Microservices Communication
- Log Aggregation,Processing and Monitoring
- Event Sourcing and CQRS
- Messaging
- Stream Processing



What the Differences Between Brokers?

	RabbitMQ	**Kafka**
Acknowledgment Handling	Messages exist in queues until they are acknowledged by consumers.	Kafka stores messages until consumers acknowledge receipt, ensuring reliable delivery.
Message Storage	Does not keep any message after acknowledge.	Keeps messages in logs. It is possible to reread the message.
Delivery guarantees	at-least-once, at-most-once.	at-least-once, exactly-once, effectively-once.
Ordering	FIFO, but messages can be prioritised or delayed.	FIFO within each partition, but does not guarantee order.
Broker-consumer model	Passive consumer, Active broker model.	Active consumer, Passive broker.

What the Differences Between Brokers?

	RabbitMQ	Kafka
Acknowledgment Handling	Messages exist in queues until they are acknowledged by **consumers**.	Messages exist in patritions until they are acknowledged by **producer**.
Message Storage	Does not keep any message after acknowledge.	Keeps messages in logs. Is is possible to reread the message.
Delivery guarantees	at-least-once, at-most-once.	at-least-once, exactly-once, effectively-once.
Ordering	FIFO, but messages can be prioritised or delayed.	FIFO within each partition, but does not guarantee order.
Broker-consumer model	Passive consumer, Active broker model.	Active consumer, Passive broker.

What the Differences Between Brokers?

	RabbitMQ	Kafka
Acknowledgment Handling	Messages exist in queues until they are acknowledged by consumers.	Kafka stores messages until consumers acknowledge receipt, ensuring reliable delivery.
Message Storage	Does not keep any message after acknowledge.	Keeps messages in logs. Is is possible to reread the message.
Delivery guarantees	at-least-once, at-most-once.	at-least-once, exactly-once, effectively-once.
Ordering	FIFO, but messages can be prioritised or delayed.	FIFO within each partition, but does not guarantee order.
Broker-consumer model	Passive consumer, Active broker model.	Active consumer, Passive broker.

What the Differences Between Brokers?

	RabbitMQ	Kafka
Acknowledgment Handling	Messages exist in queues until they are acknowledged by consumers.	Kafka stores messages until consumers acknowledge receipt, ensuring reliable delivery.
Message Storage	Does not keep any message after acknowledge.	Keeps messages in logs. Is is possible to reread the message.
Delivery guarantees	**at-least-once**, **at-most-once.**	**at-least-once**, **exactly-once**, **effectively-once.**
Ordering	FIFO, but messages can be prioritised or delayed.	FIFO within each partition, but does not guarantee order.
Broker-consumer model	Passive consumer, Active broker model.	Active consumer, Passive broker.

What the Differences Between Brokers?

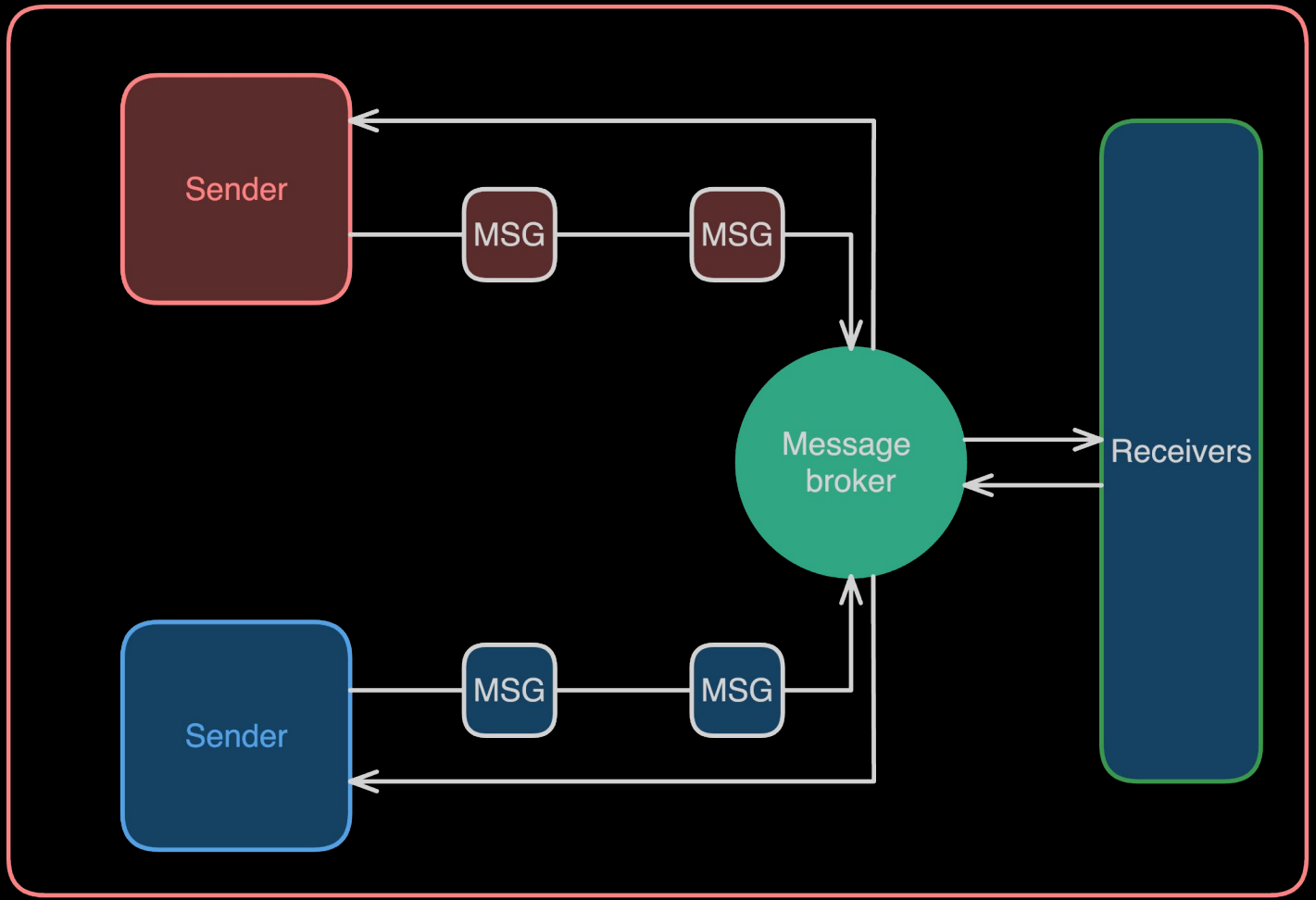
	RabbitMQ	**Kafka**
Acknowledgment Handling	Messages exist in queues until they are acknowledged by consumers.	Kafka stores messages until consumers acknowledge receipt, ensuring reliable delivery.
Message Storage	Does not keep any message after acknowledge.	Keeps messages in logs. Is is possible to reread the message.
Delivery guarantees	at-least-once, at-most-once.	at-least-once, exactly-once, effectively-once.
Ordering	**FIFO**, but messages **can be prioritised** or delayed.	**FIFO** within each partition, but **does not guarantee order**.
Broker-consumer model	Passive consumer, Active broker model.	Active consumer, Passive broker.

What the Differences Between Brokers?

	RabbitMQ	**Kafka**
Acknowledgment Handling	Messages exist in queues until they are acknowledged by consumers.	Kafka stores messages until consumers acknowledge receipt, ensuring reliable delivery.
Message Storage	Does not keep any message after acknowledge.	Keeps messages in logs. Is is possible to reread the message.
Delivery guarantees	at-least-once, at-most-once.	at-least-once, exactly-once, effectively-once.
Ordering	FIFO, but messages can be prioritised or delayed.	FIFO within each partition, but does not guarantee order.
Broker-consumer model	**Passive** consumer, **Active** broker model.	**Active** consumer, **Passive** broker.

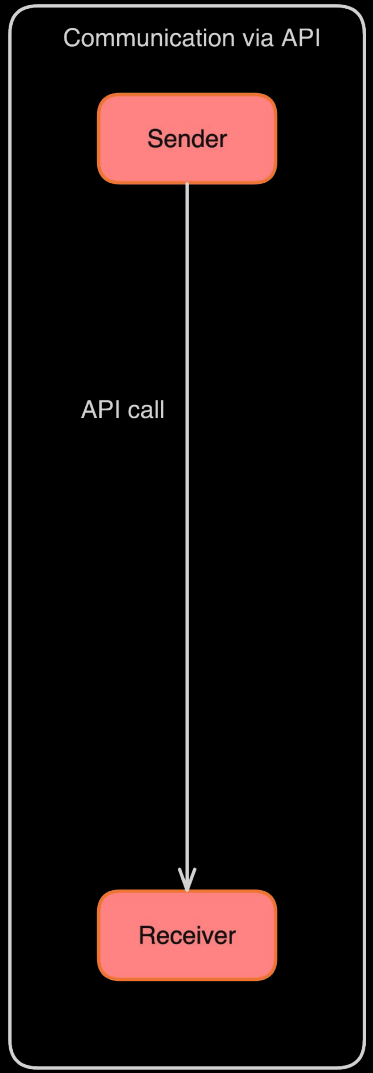
Why we use Message Brokers?

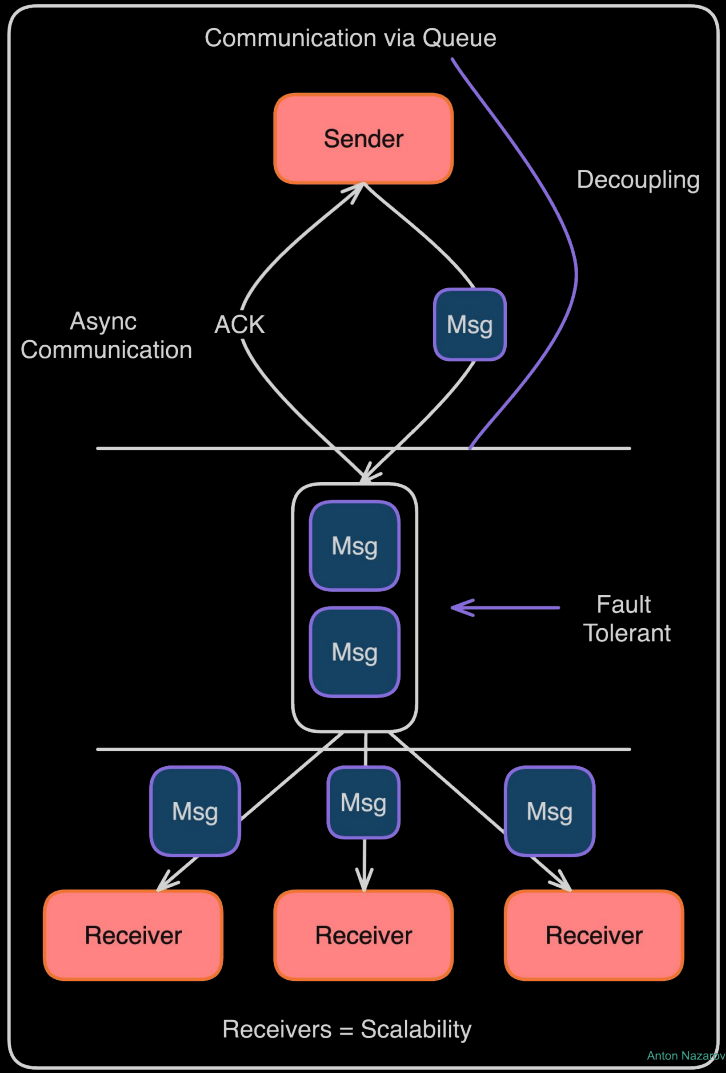
Decoupling and Loose Coupling



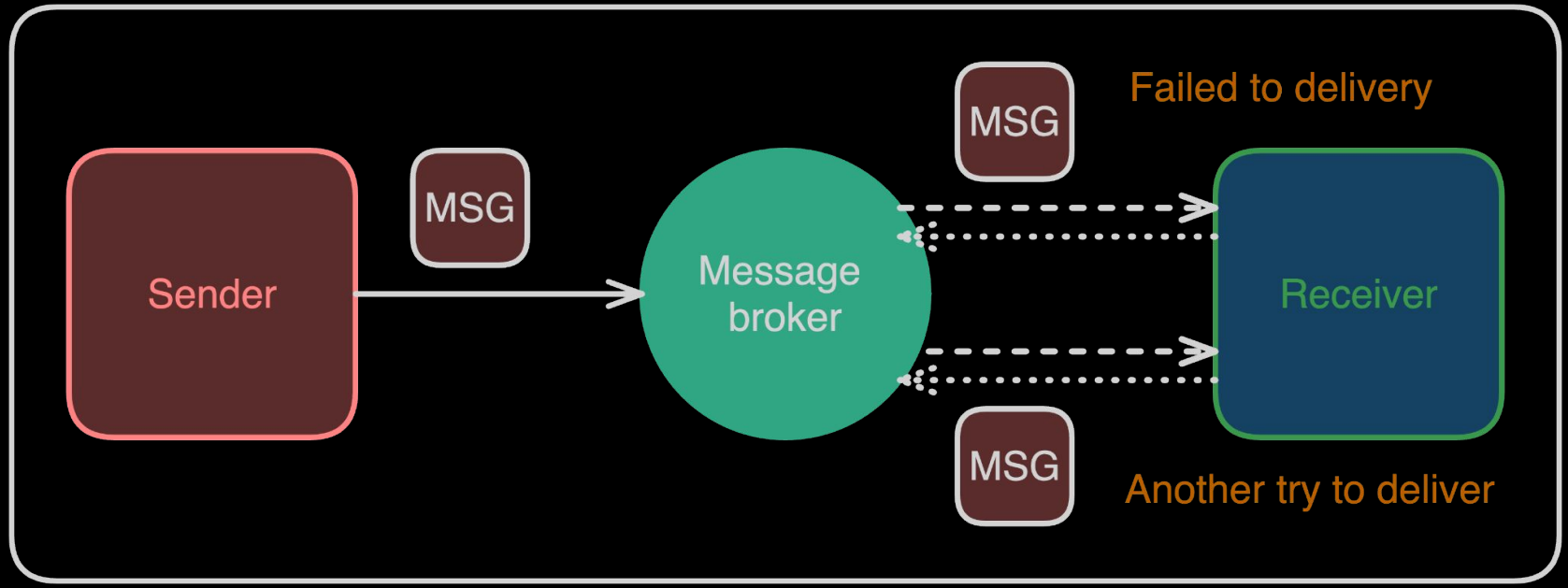
Asynchronous Communication

1. API is slow (operates synchronously).
2. APIs do not guarantee message ordering between requests.
3. Direct API integration can lead to tight coupling between services.
4. APIs typically do not provide built-in mechanisms for message durability and persistence.
5. API calls are prone to network failures, server timeouts, and other transient errors.
6. …

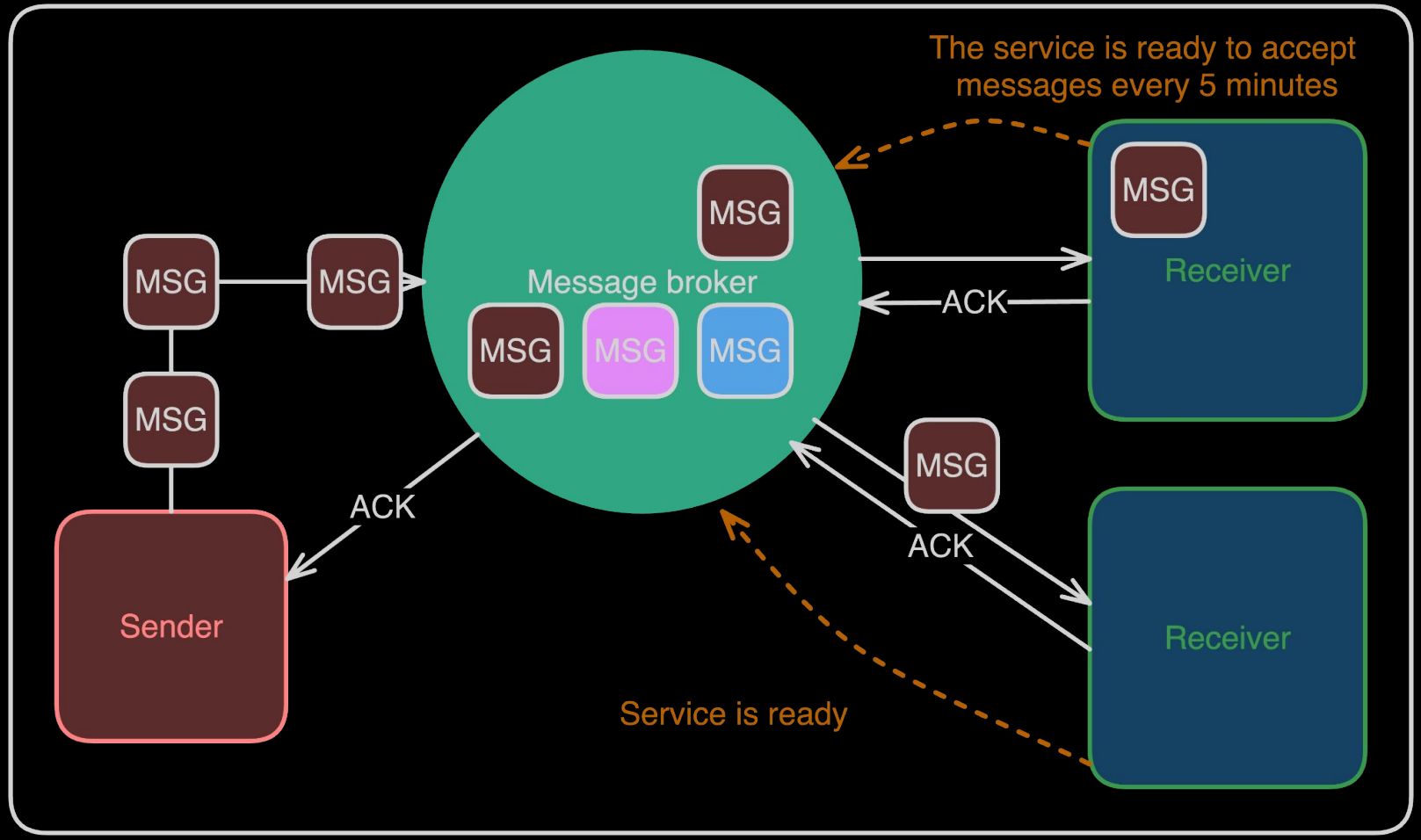




Reliability and Delivery Guarantees



Scalability and Load Balancing



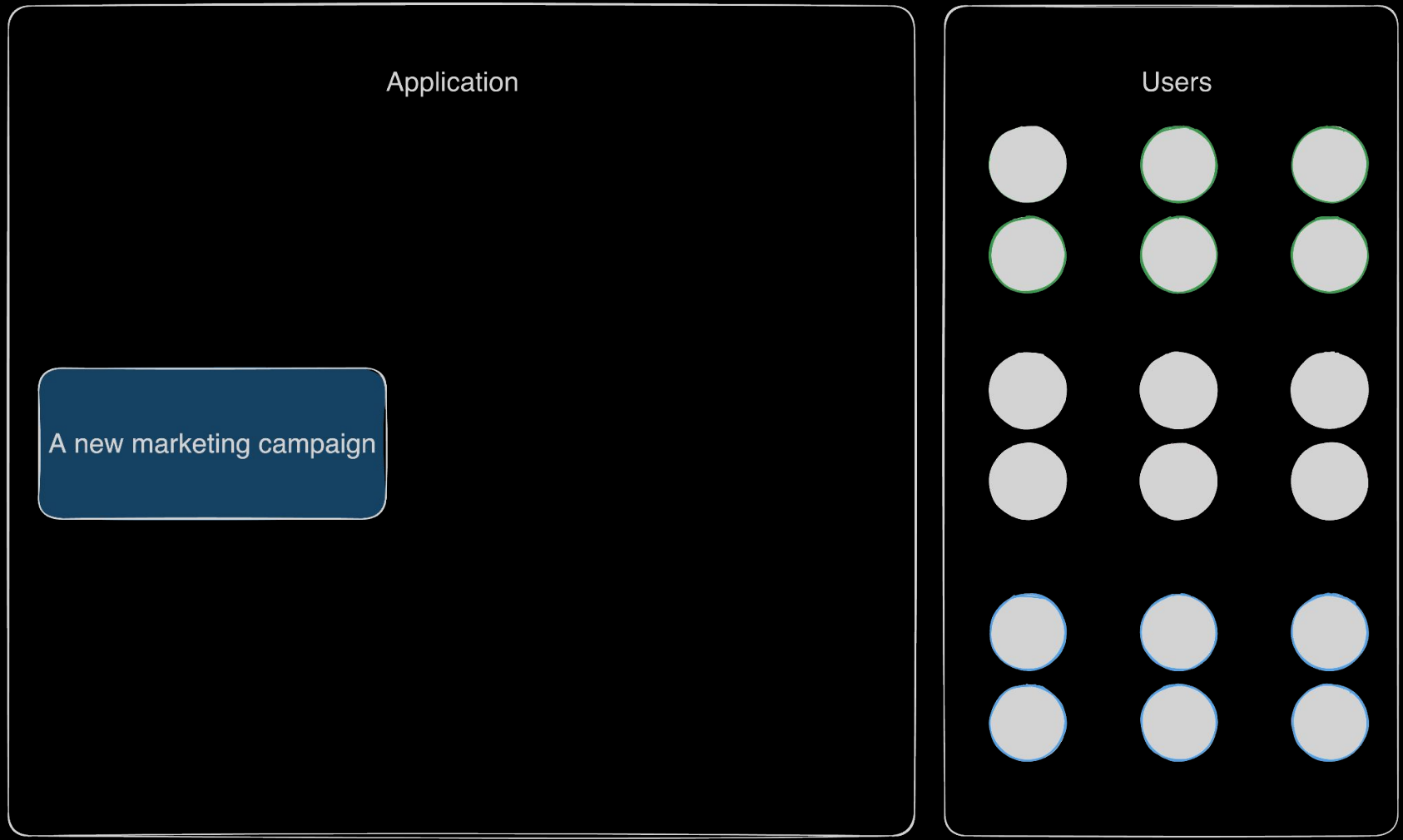
What are the disadvantages of implementing a message broker?

What are the disadvantages of implementing a message broker?

- Potential performance bottleneck.
- Potential single Point of failure.
- Additional operational complexity - Debugging / Learning curve / Maintenance.
- Increasing the volume of traffic.

A case study

- The task requires to send 1 million messages.
- Users should get their messages within a time period of 1-2pm the same day.
- Each message is personalised.

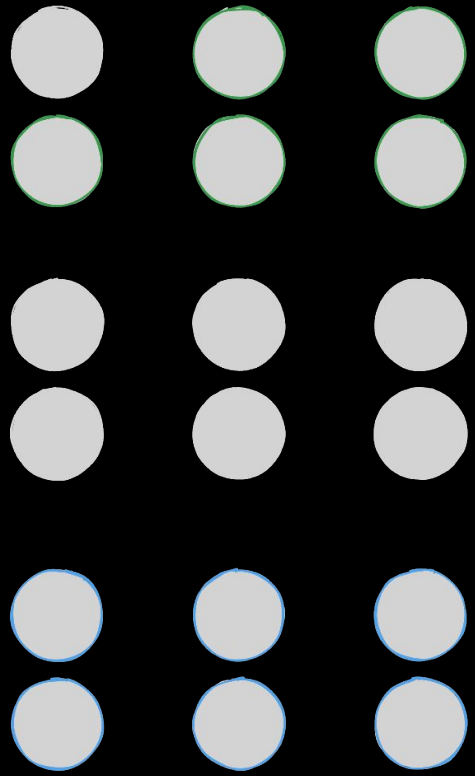


Anton Nazarov

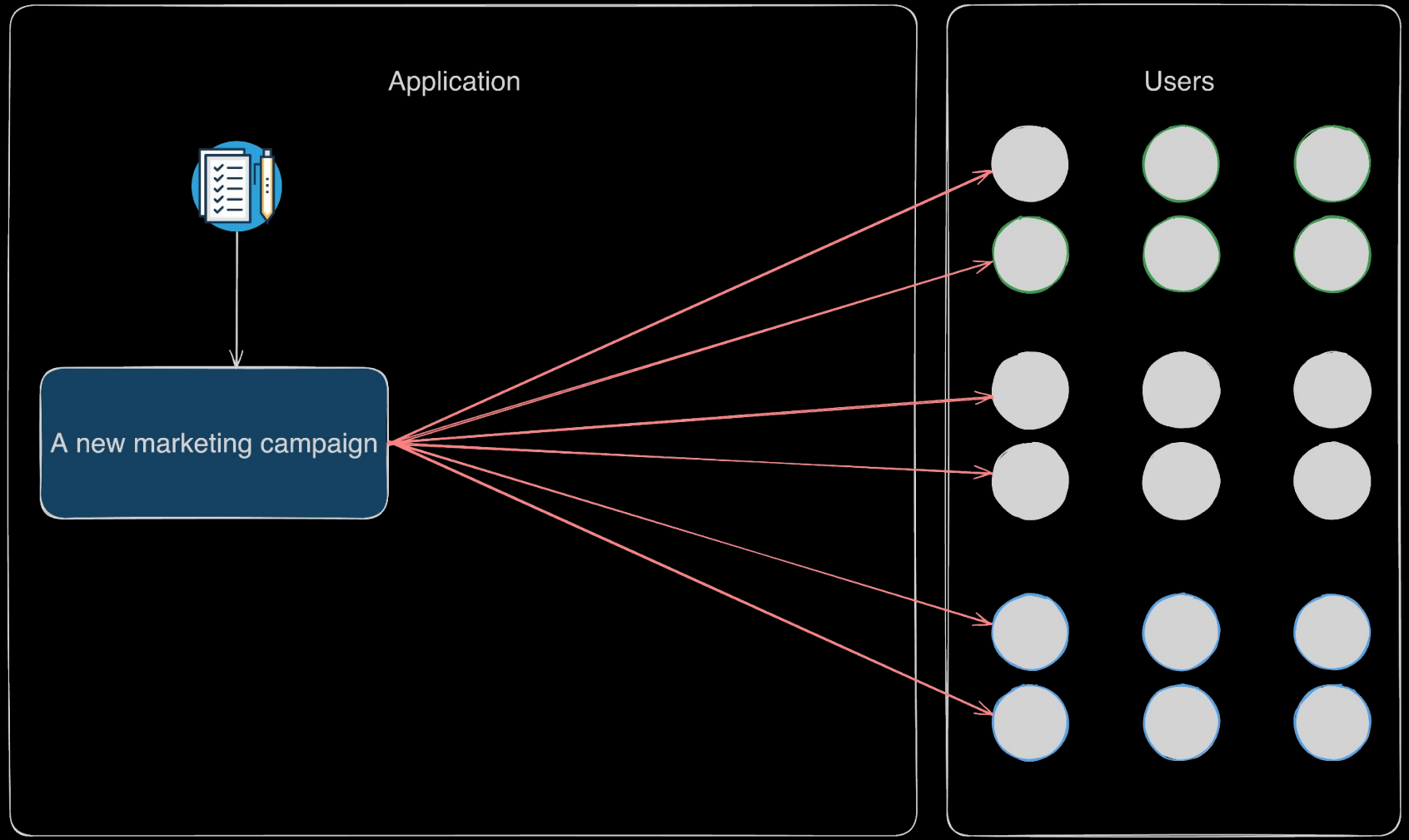
Application

A new marketing campaign

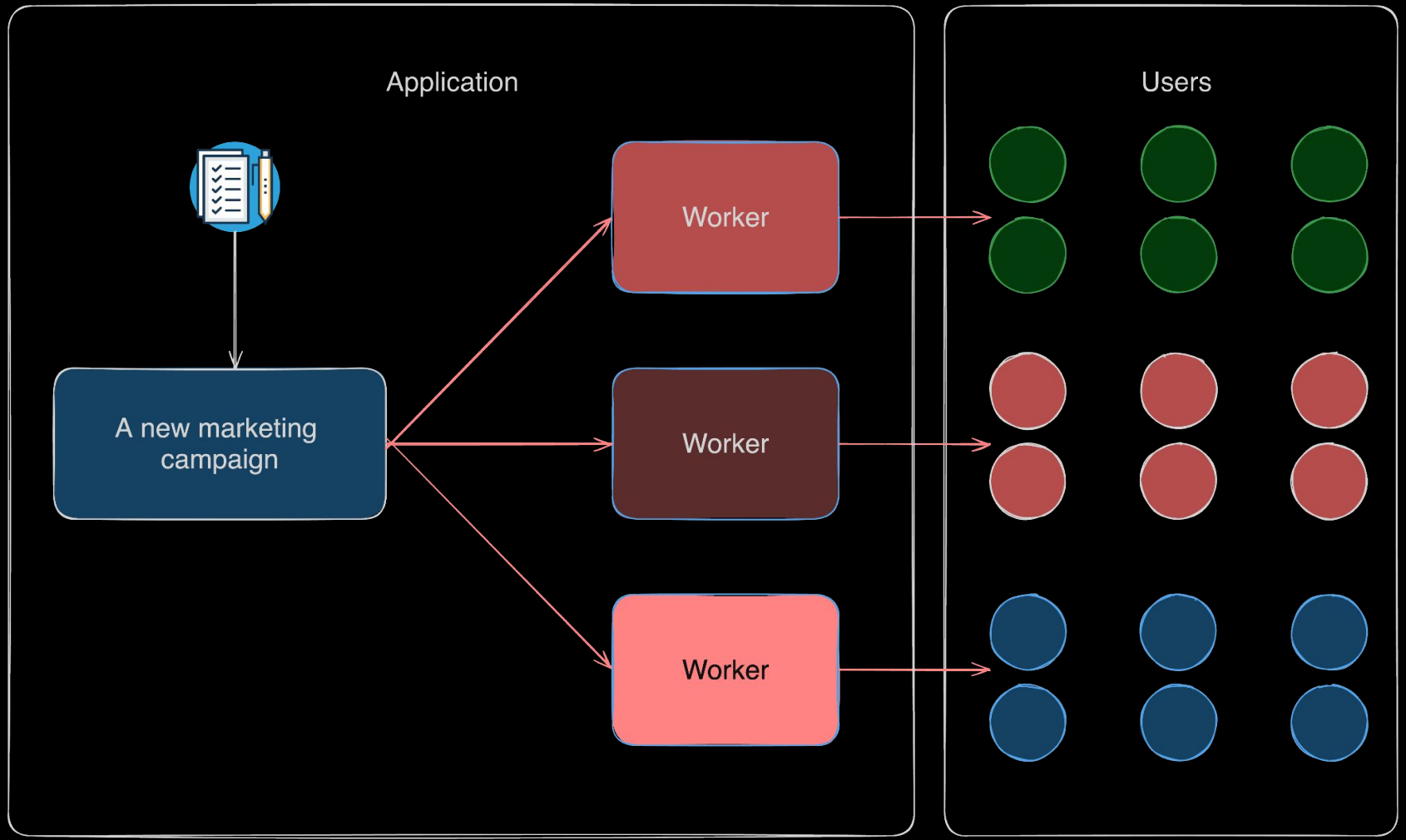
Users



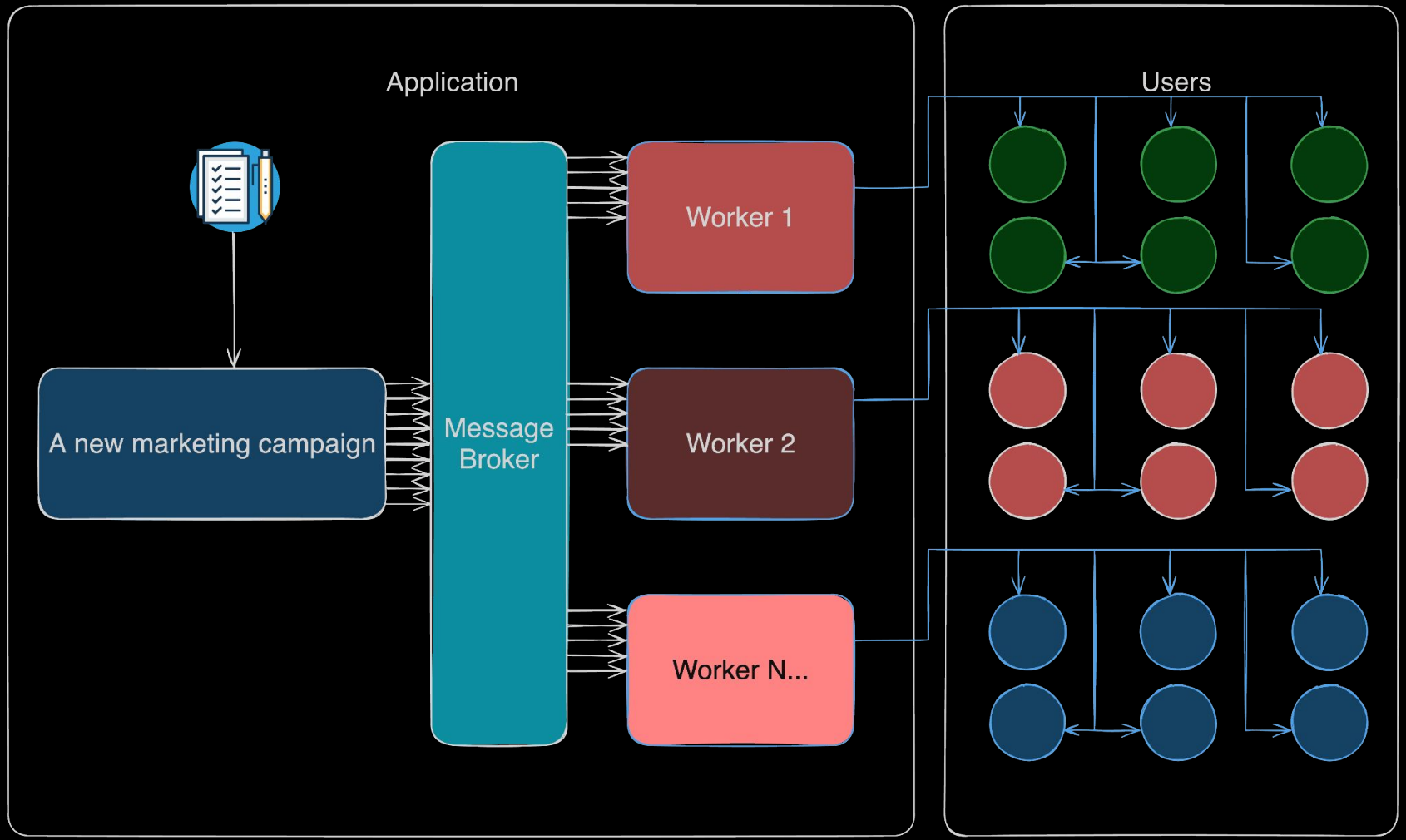
Anton Nazarov



Anton Nazarov



Anton Nazarov



Anton Nazarov

How to start?

Documentation:

https://www.rabbitmq.com/docs

https://kafka.apache.org/documentation/

https://activemq.apache.org/components/classic/documentation/

Docker Hub:

https://hub.docker.com/_/rabbitmq

https://hub.docker.com/r/apache/kafka

https://hub.docker.com/r/apache/activemq-classic

How to start?

Developer tools:

https://www.rabbitmq.com/client-libraries/devtools

https://cwiki.apache.org/confluence/display/KAFKA/Clients

https://activemq.apache.org/components/classic/documentation/cross-language-clients

Books:

https://www.oreilly.com/library/view/rabbitmq-in-action/9781935182979/

https://www.oreilly.com/library/view/rabbitmq-in-depth/9781617291005/

https://www.oreilly.com/library/view/kafka-the-definitive/9781492043072/

Whats next? Enjoy!

Online simulations:

https://tryrabbitmq.com/

https://softwaremill.com/kafka-visualisation/

Thank you

Anton Nazarov

Tony Nazarov

LinkedIn:
https://linkedin.com/in/tonynazarov

Email:
tonynazarov.nz@gmail.com

Presentations:
https://linktr.ee/tonynazarov.nz

SCAN ME

My Contacts: