

# Concurrent Programming Using The Disruptor

Trisha Gee, Developer at LMAX

@trisha\_gee

[mechanitis.blogspot.com](http://mechanitis.blogspot.com)



**The Disruptor?**

# What I'm covering

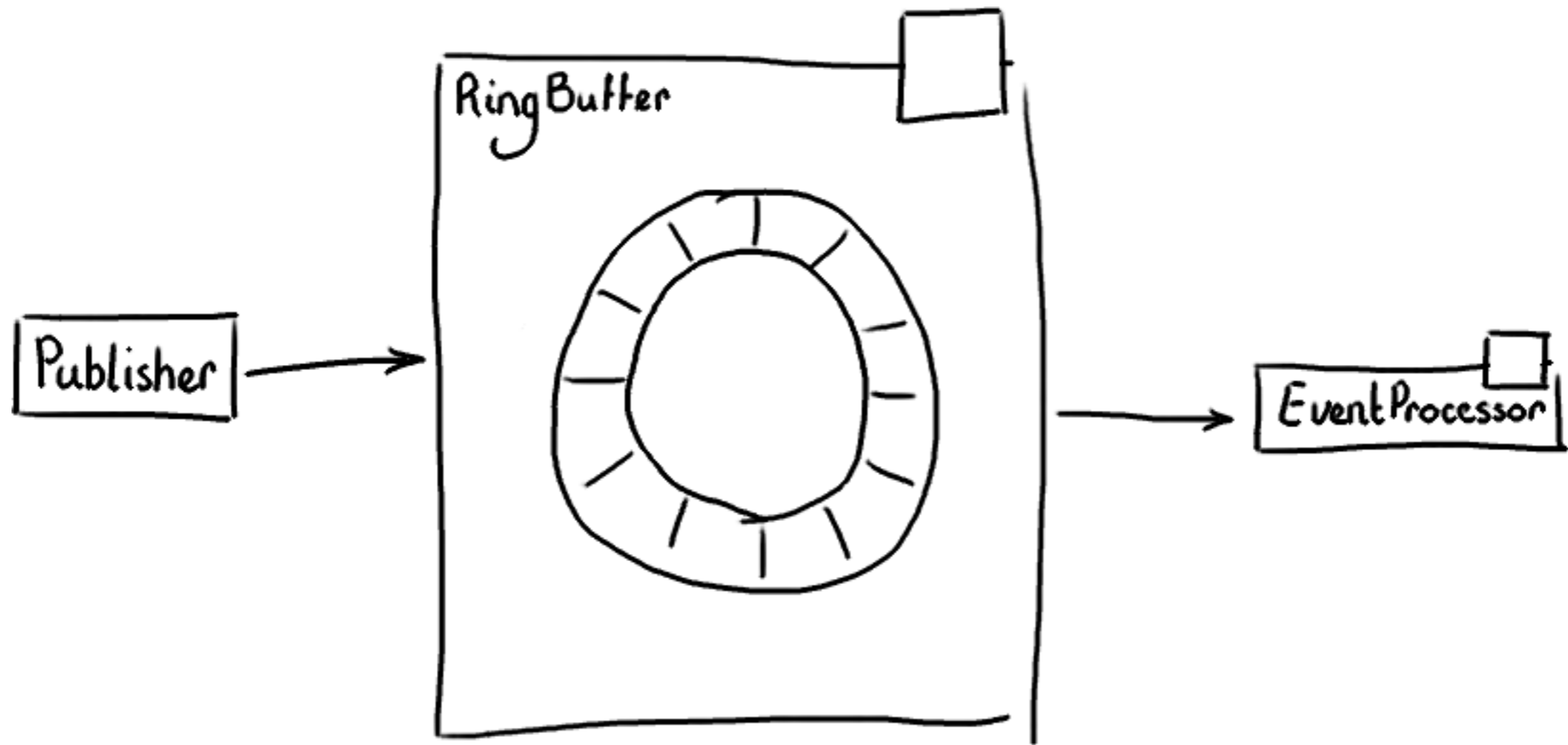
- Overview of the Disruptor
- Create your own!
- Turn it up to Eleven
- Q&A

# What is it?

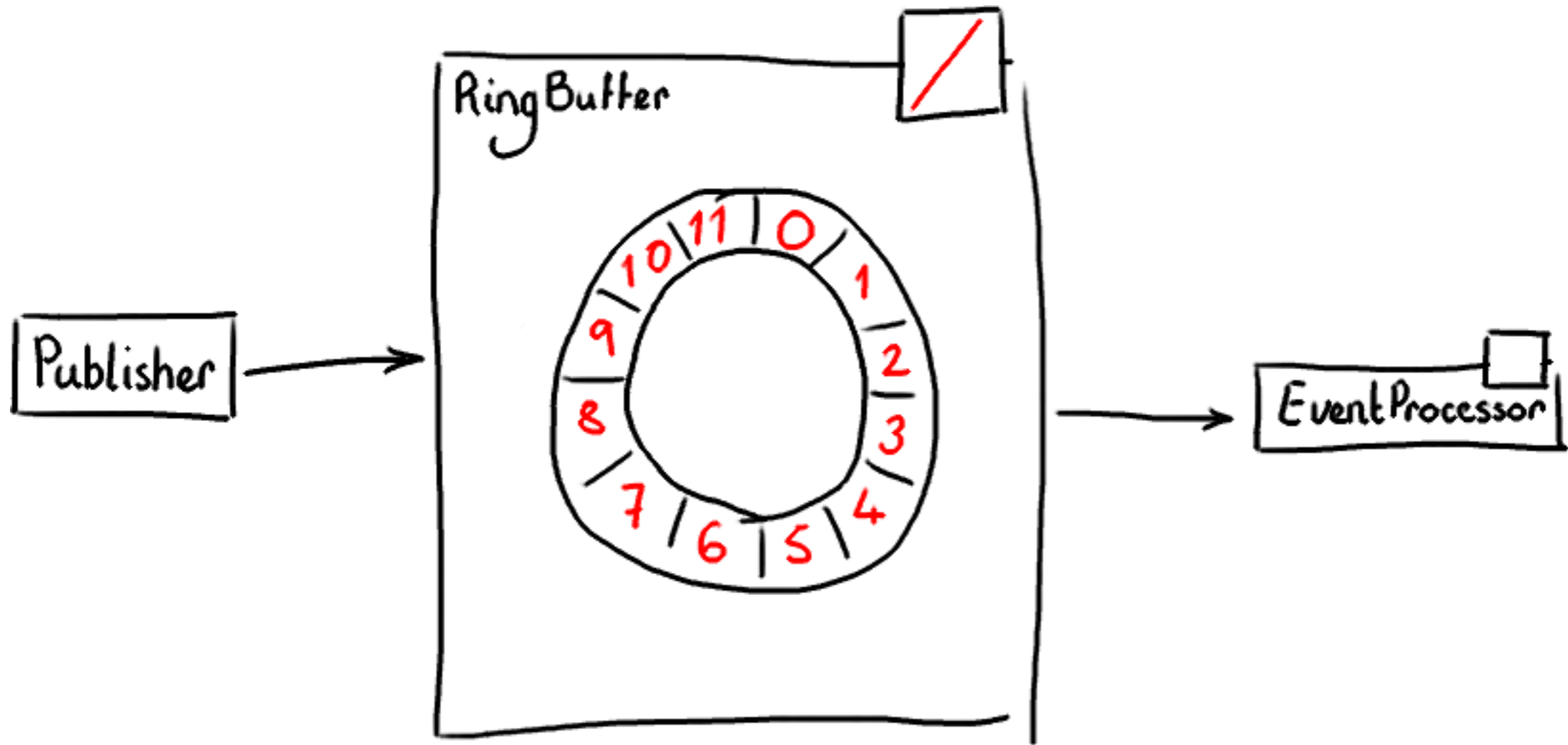
- Data structure and work flow with no contention.
- Very fast message passing.
- Allows you to go truly parallel.

So...?

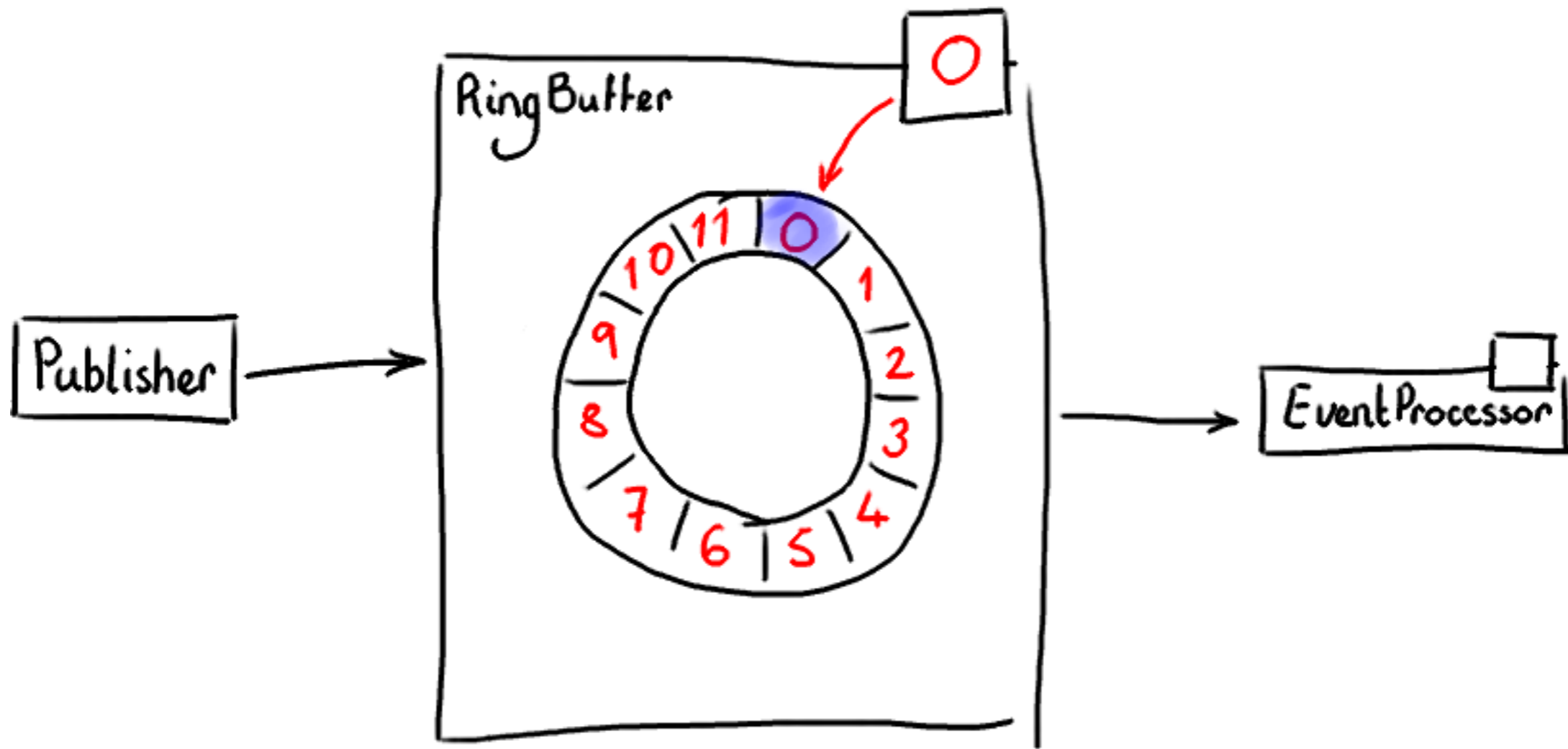
# The Magic RingBuffer



# The Magic RingBuffer

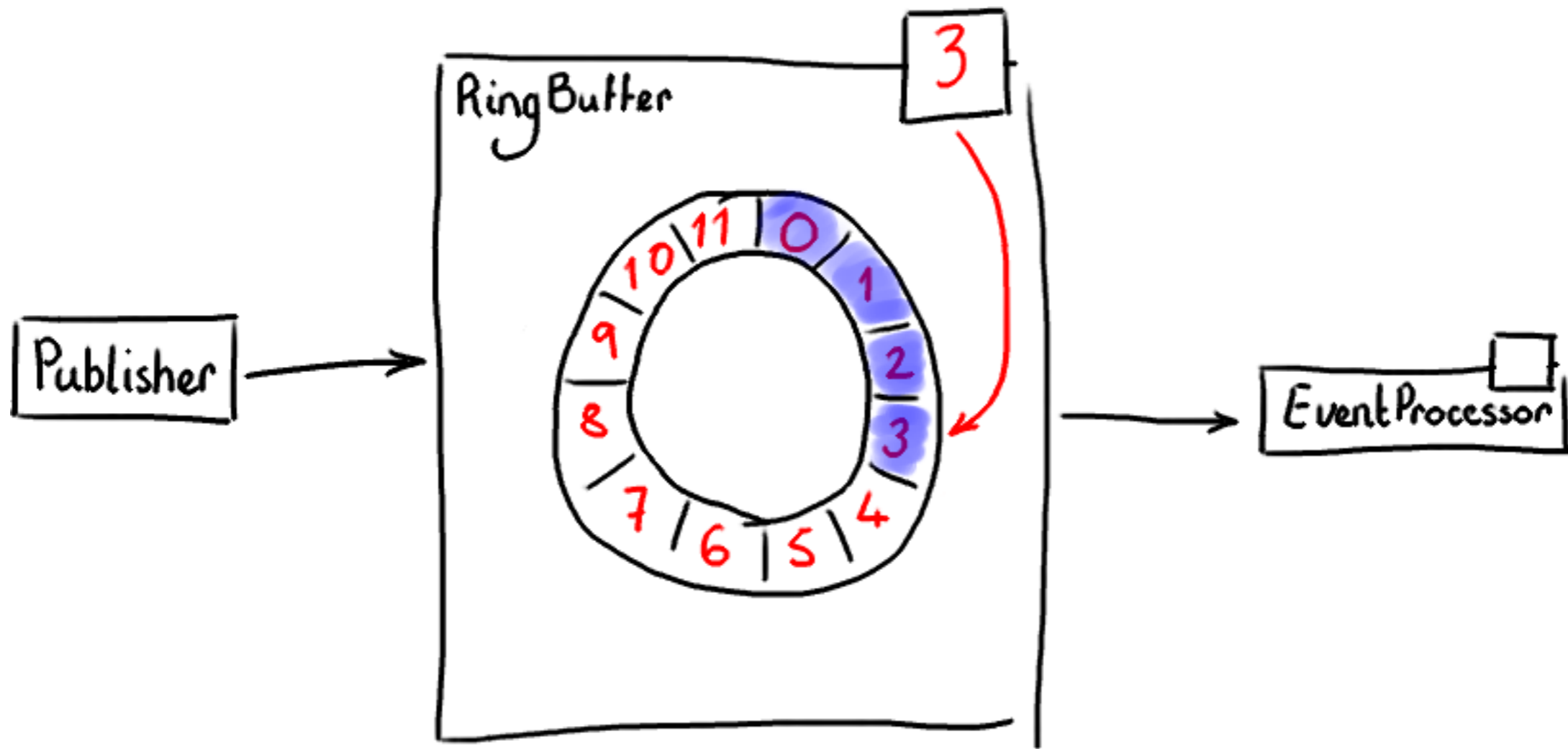


# The Magic RingBuffer

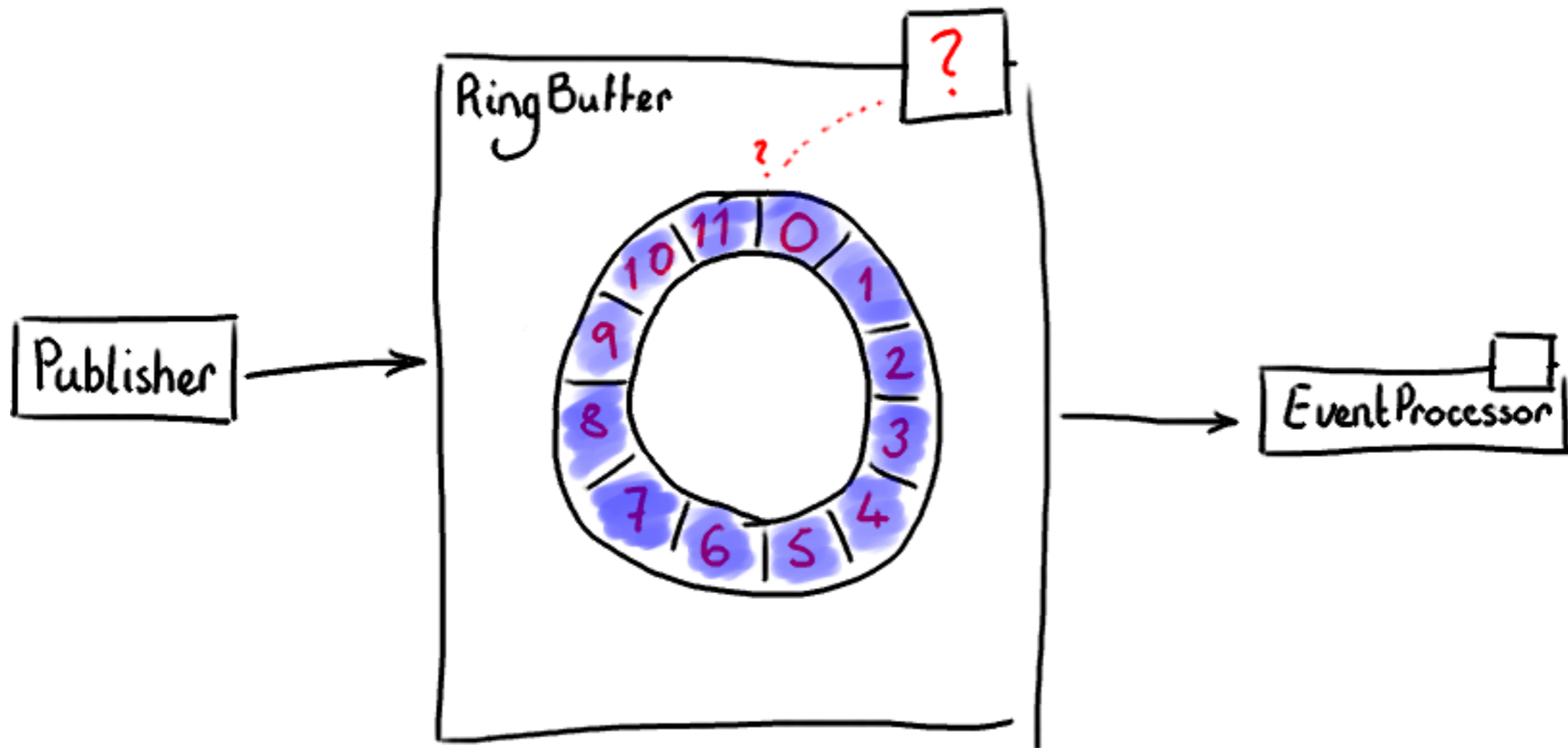




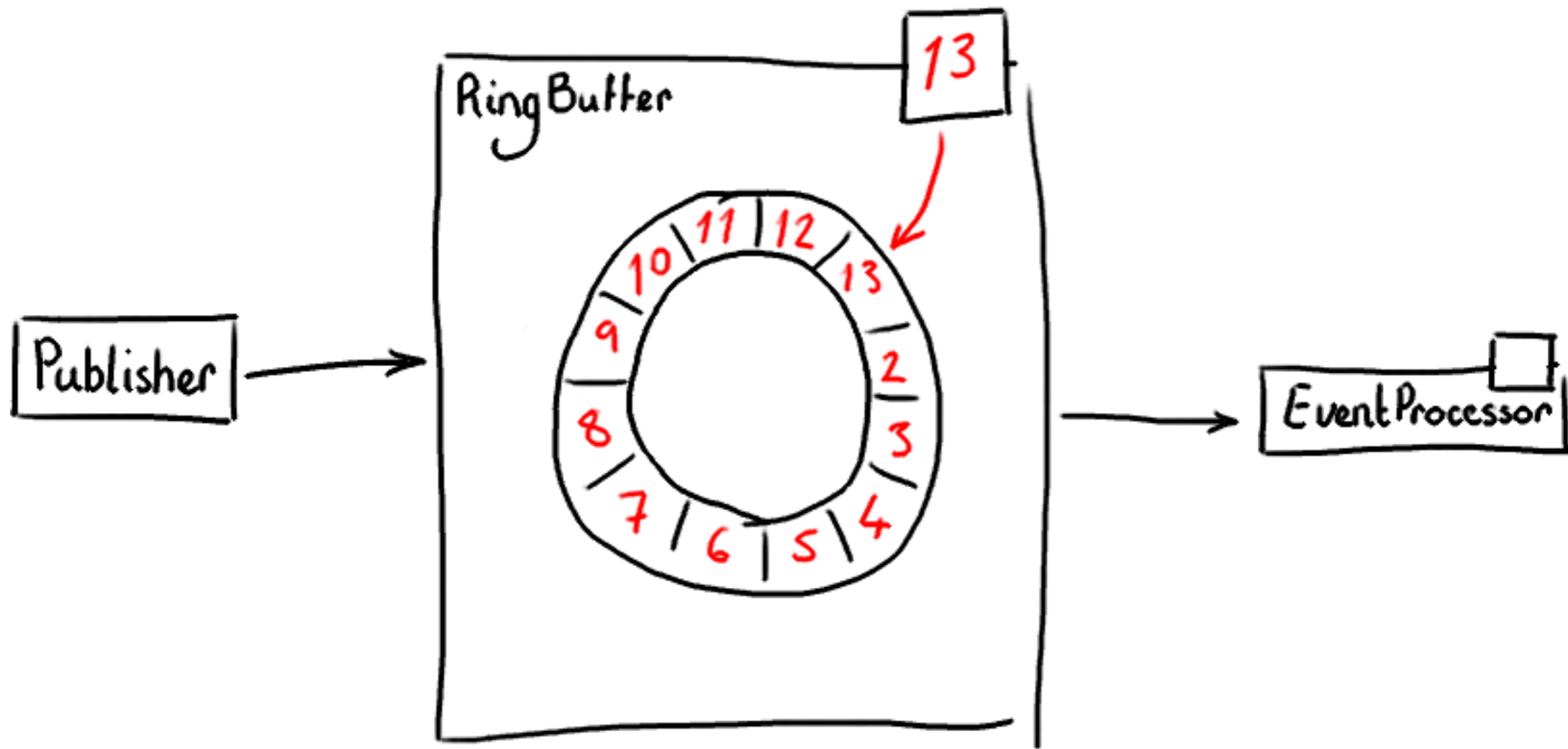
# The Magic RingBuffer



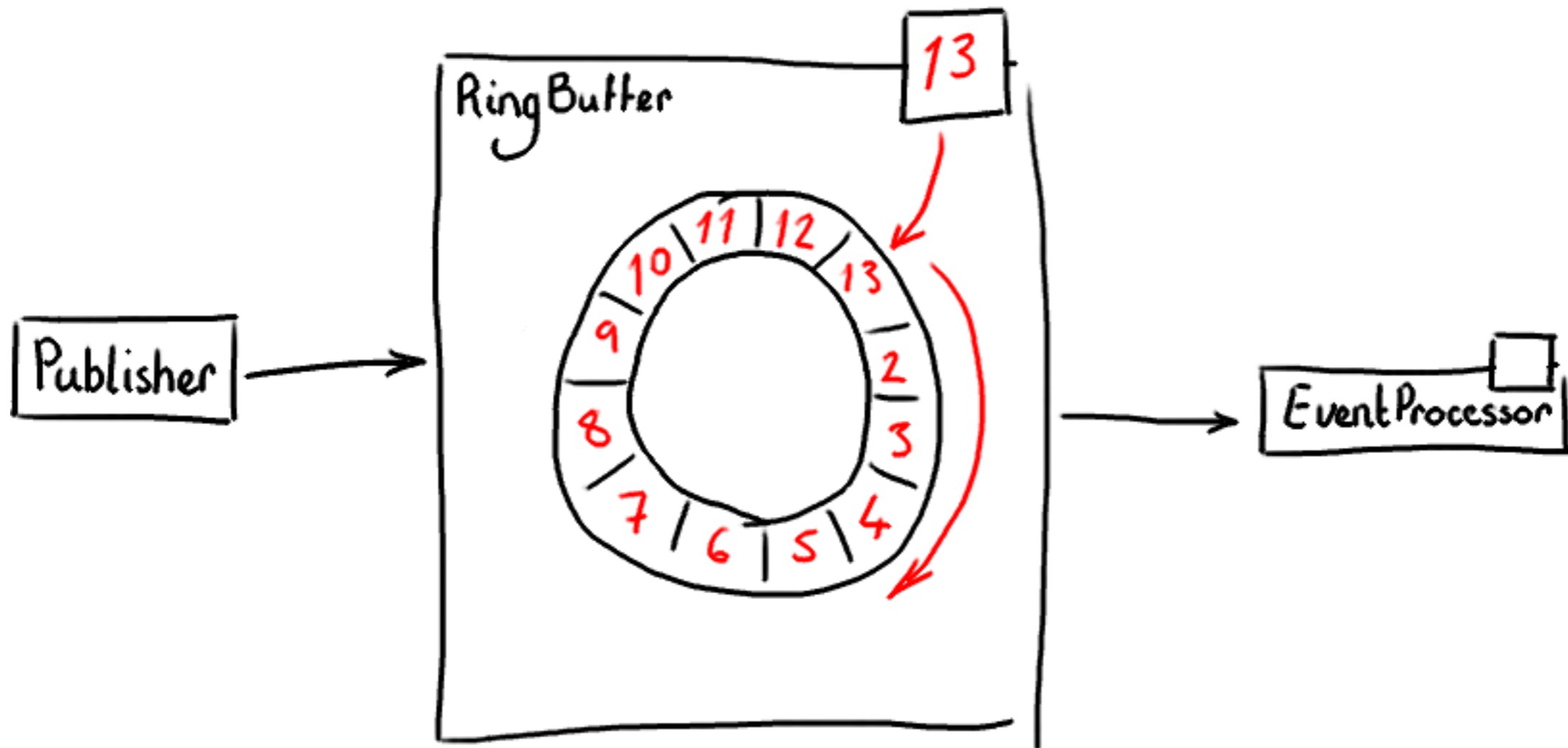
# The Magic RingBuffer



# The Magic RingBuffer



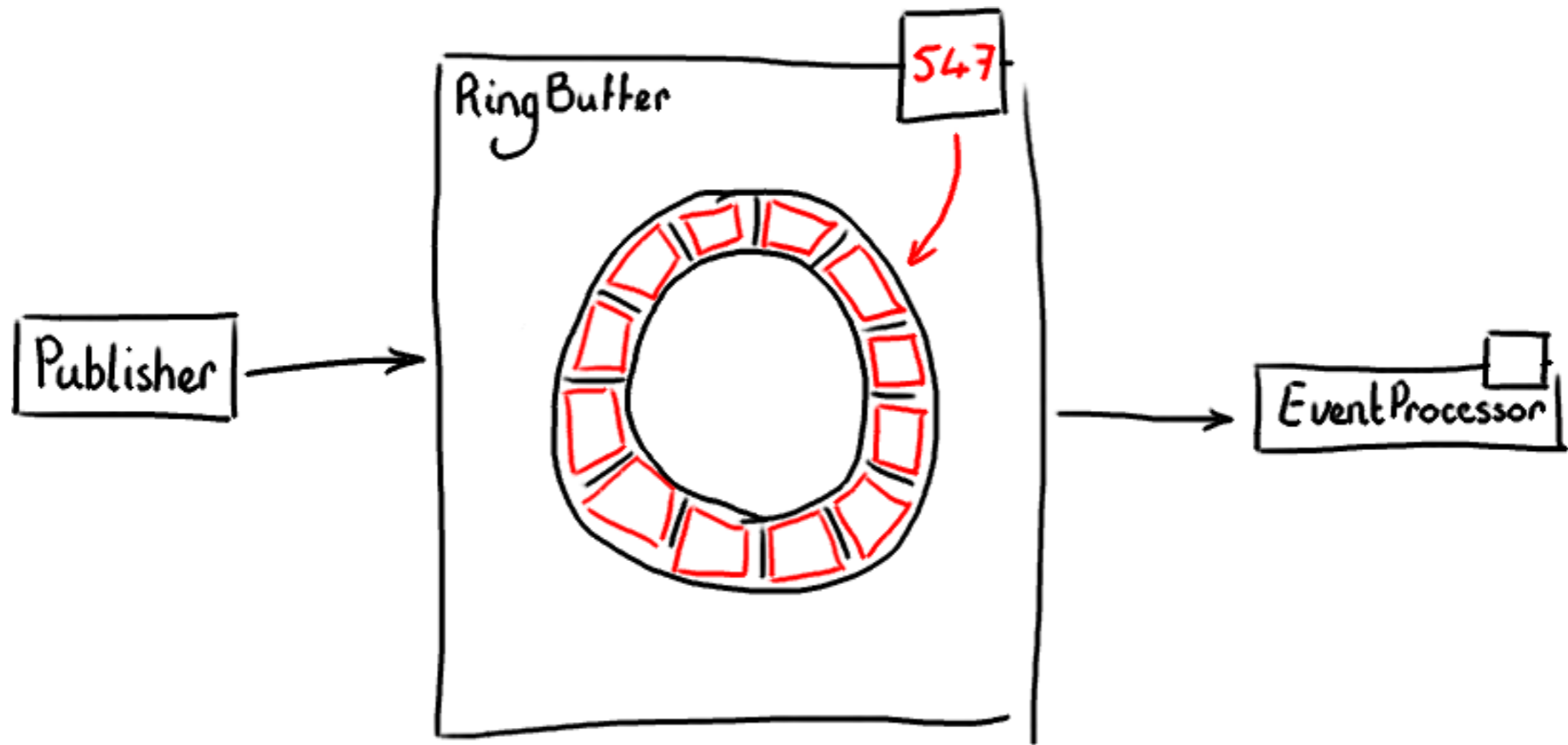
# The Magic RingBuffer



# Creating a RingBuffer

```
final RingBuffer<SimpleEvent> ringBuffer =  
    new RingBuffer<SimpleEvent>(SimpleEvent.EVENT_FACTORY,  
                                RING_BUFFER_SIZE);
```

# The Events are Buckets



# Great! I want one!

```
public class SimpleEvent {
    public static final EventFactory<SimpleEvent> EVENT_FACTORY =
        new SimpleEventFactory();

    private volatile String value;

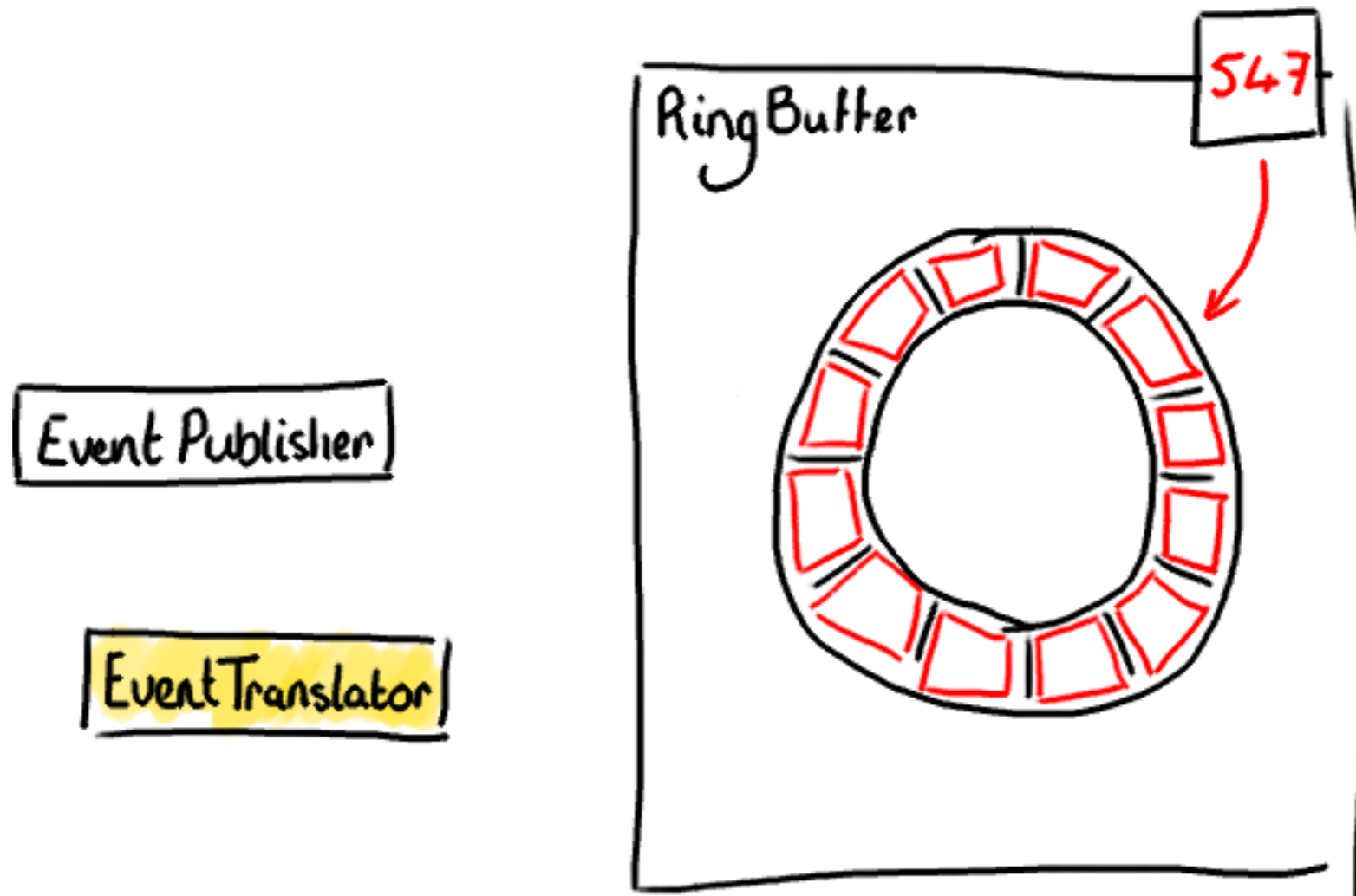
    private static class SimpleEventFactory implements EventFactory<SimpleEvent> {
        @Override
        public SimpleEvent newInstance() {
            return new SimpleEvent();
        }
    }
}
```

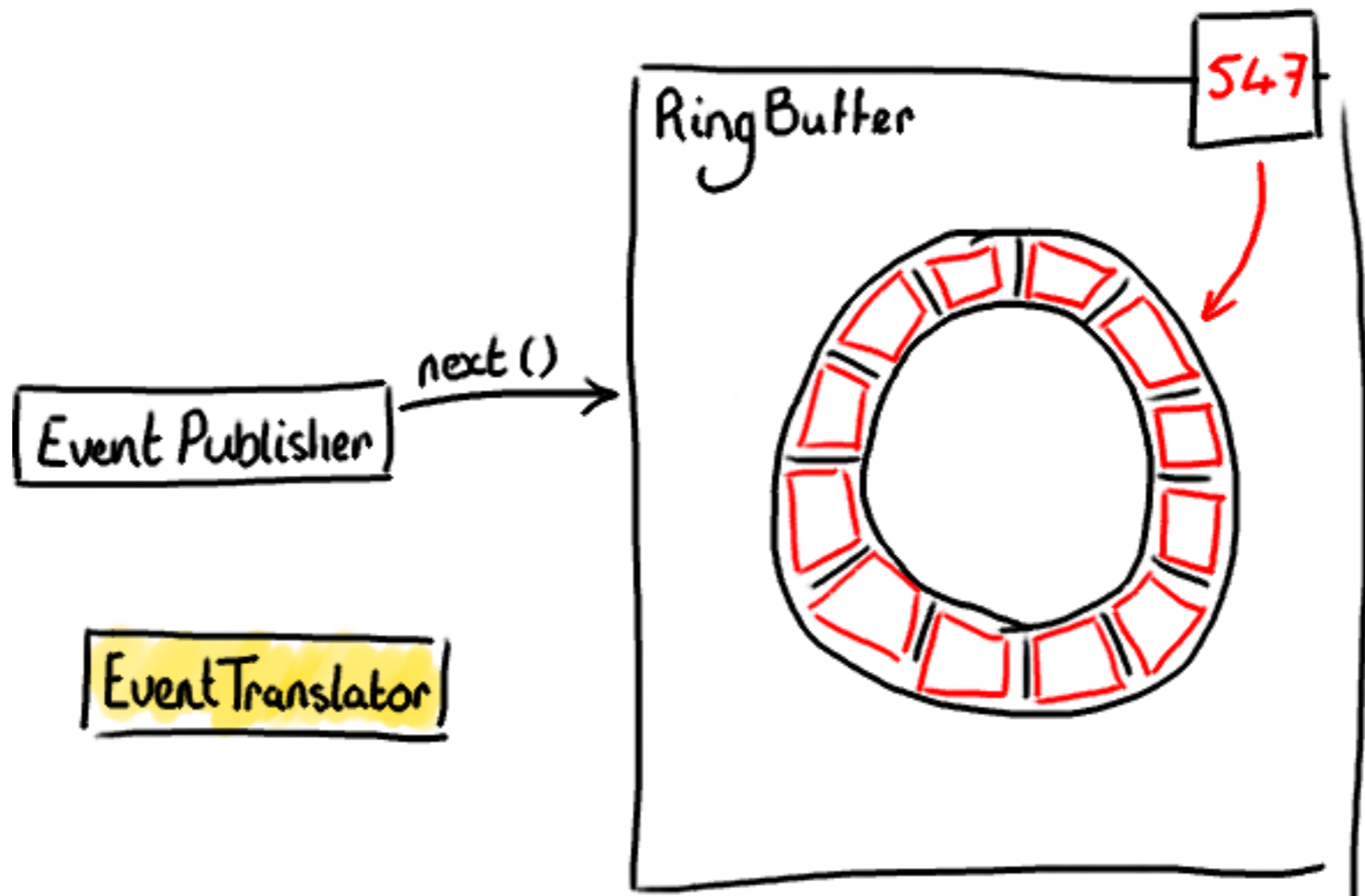
# I've got a RingBuffer!

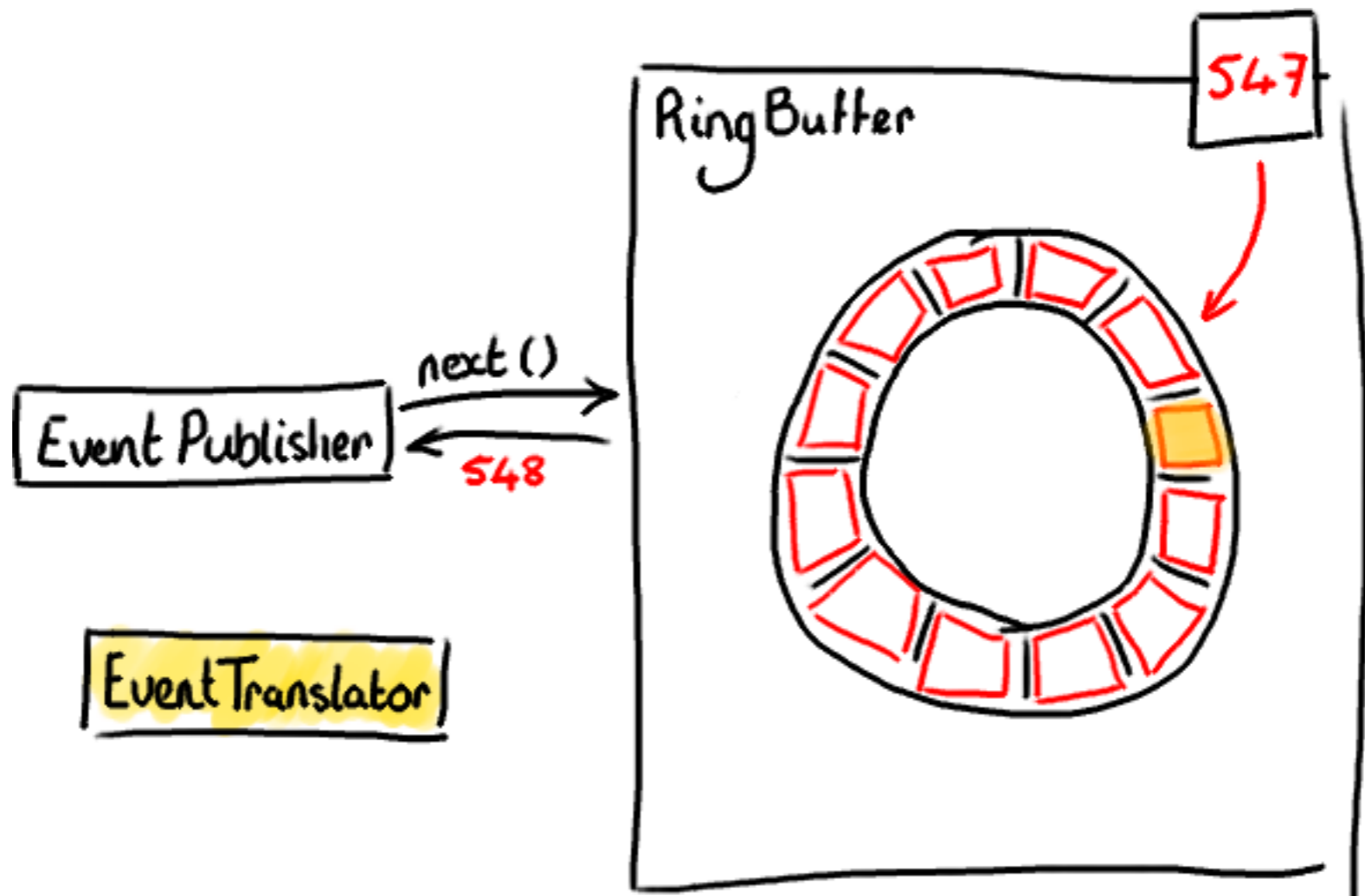
- Erm... how do I poke things into it?

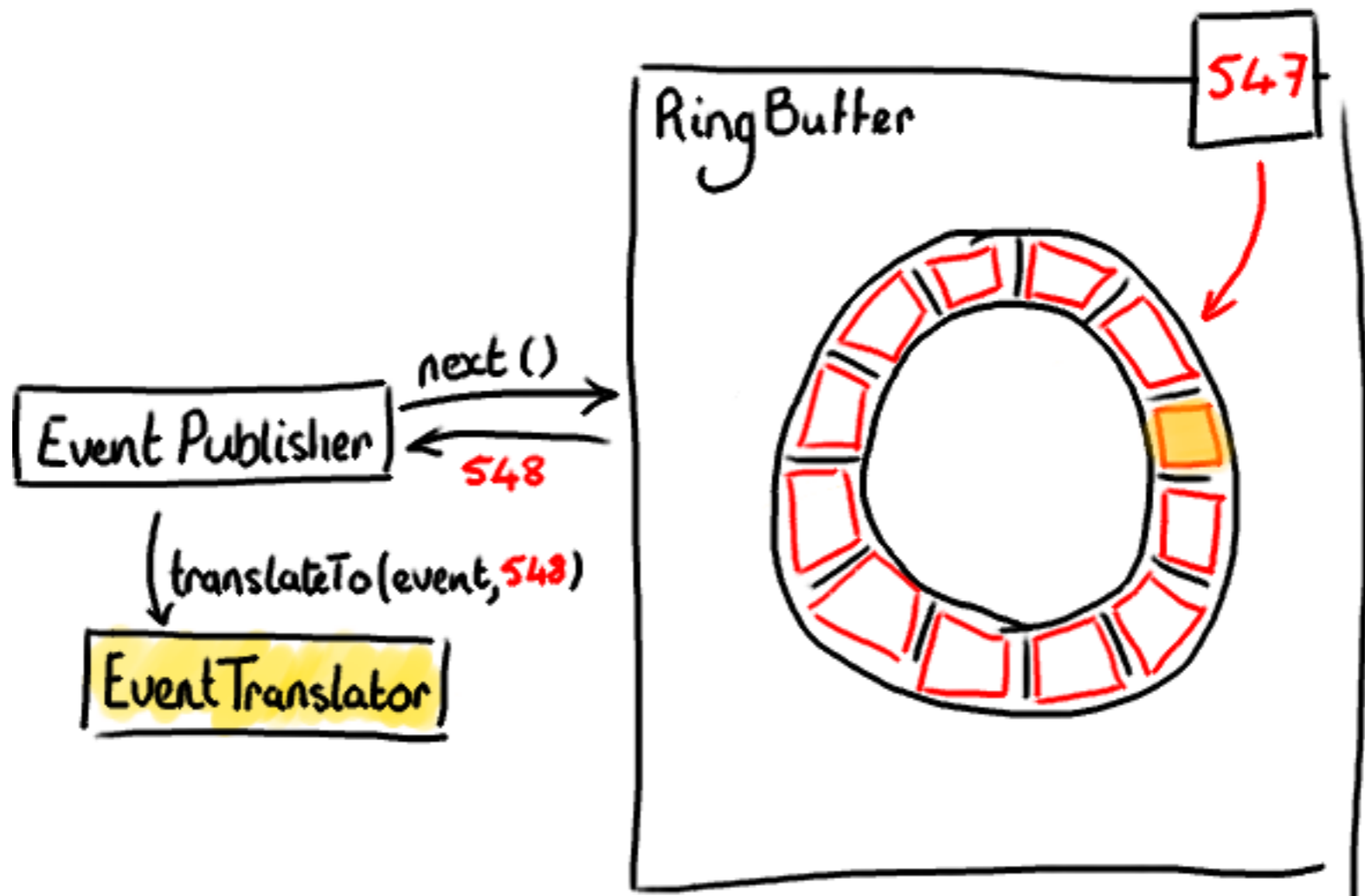


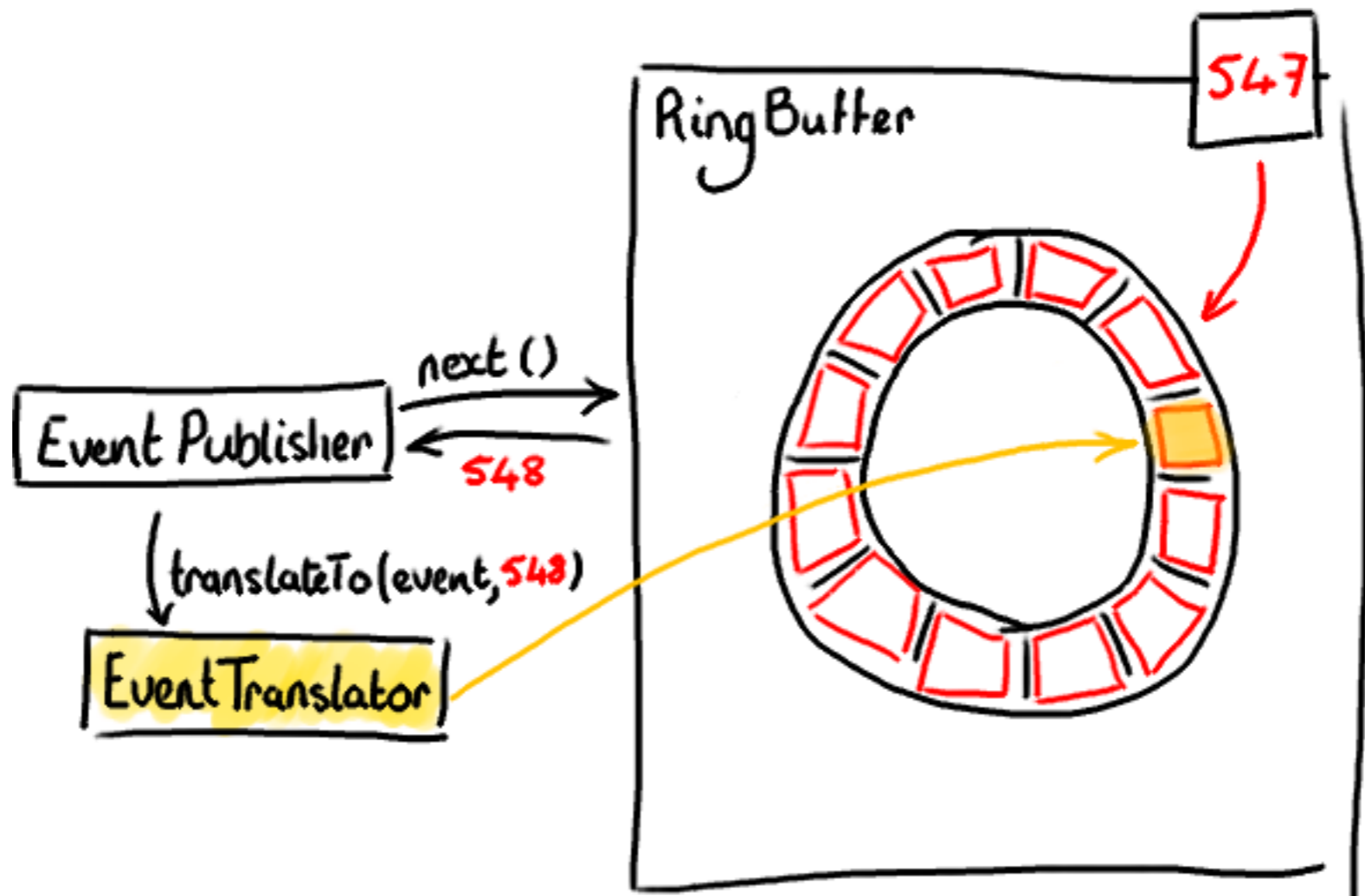
# The Publisher

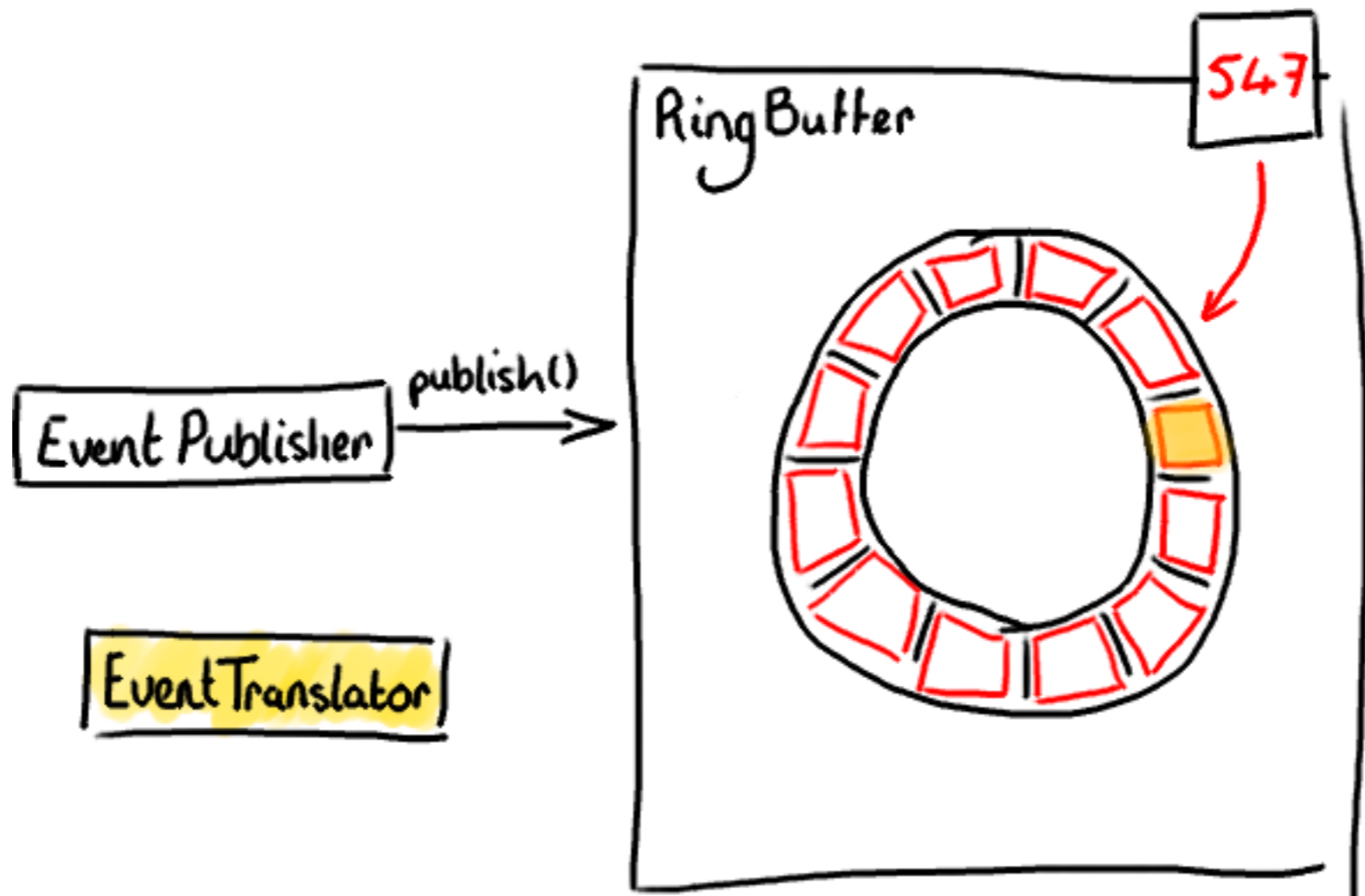


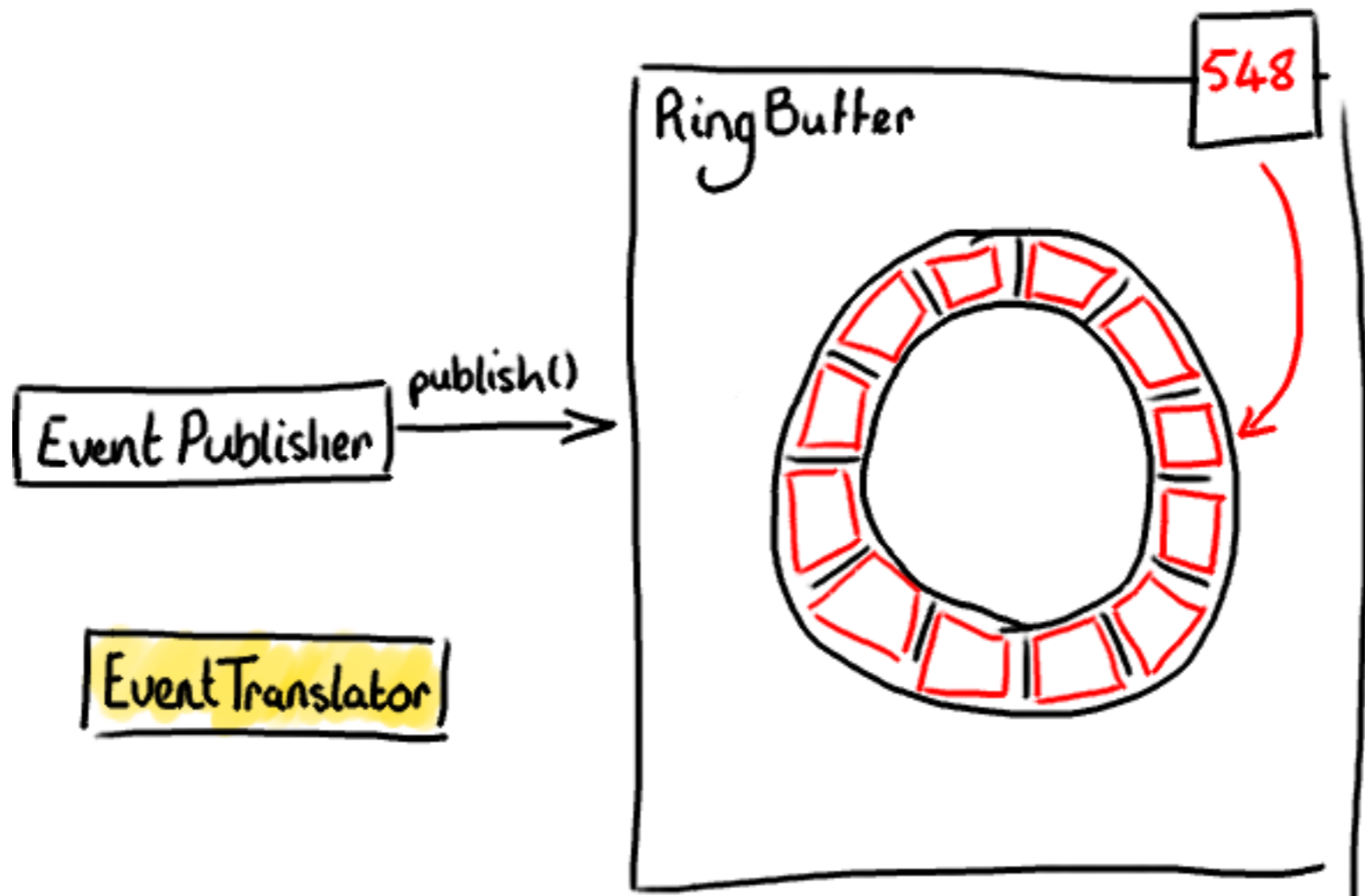












# What do I do?

```
public class SimpleEventTranslator implements  
    EventTranslator<SimpleEvent>
```

---

```
SimpleEventTranslator translator = new SimpleEventTranslator();
```

```
EventPublisher<SimpleEvent> publisher =  
    new EventPublisher<SimpleEvent>(ringBuffer);
```

```
// poke your translator here
```

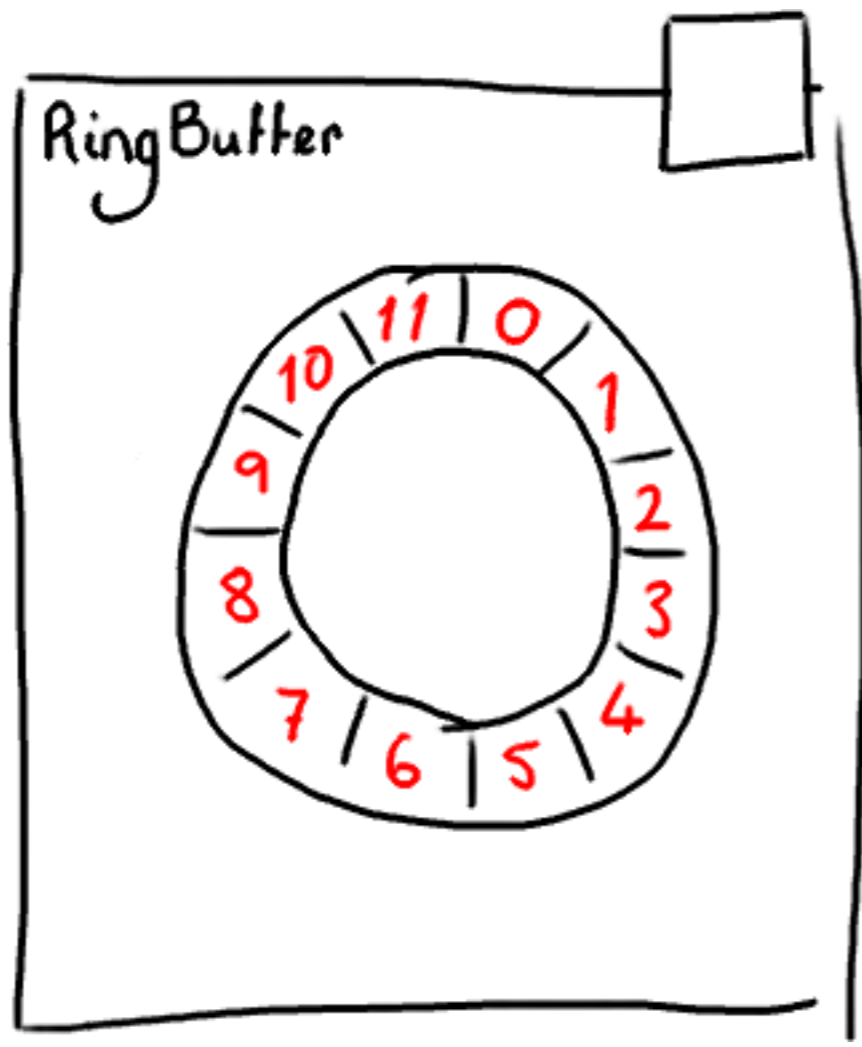
```
// ...and when you're done...
```

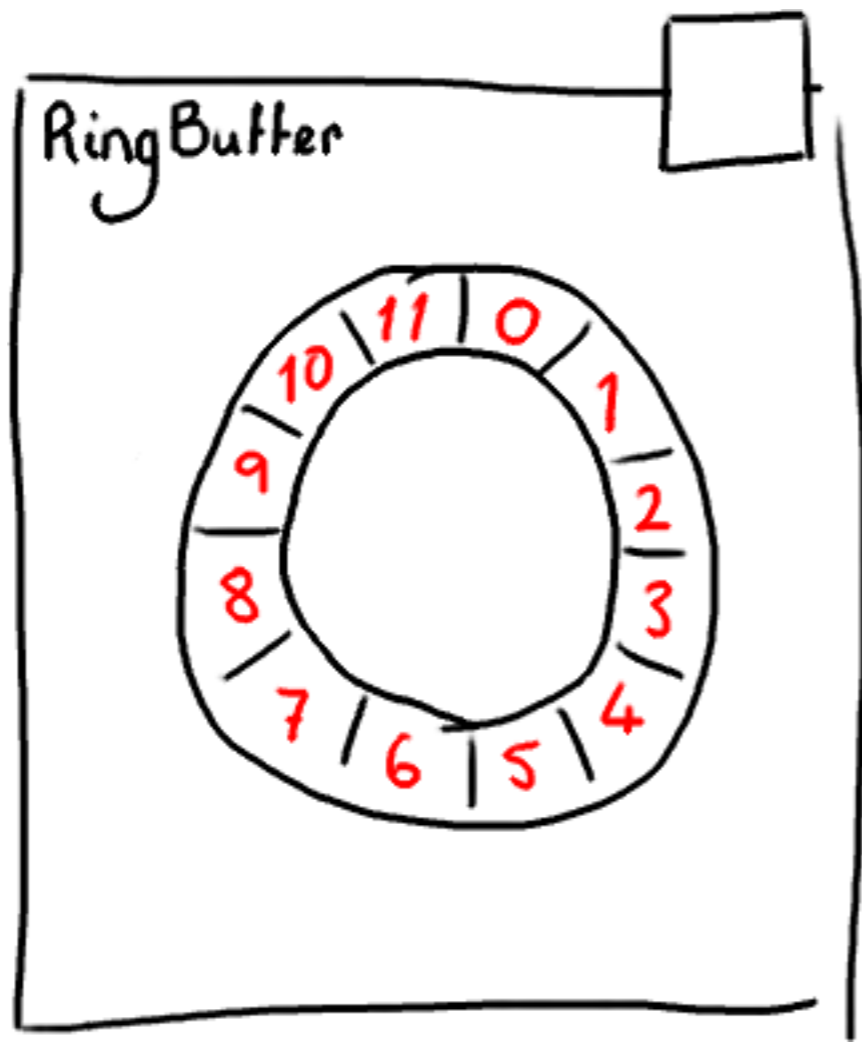
```
publisher.publishEvent(translator);
```

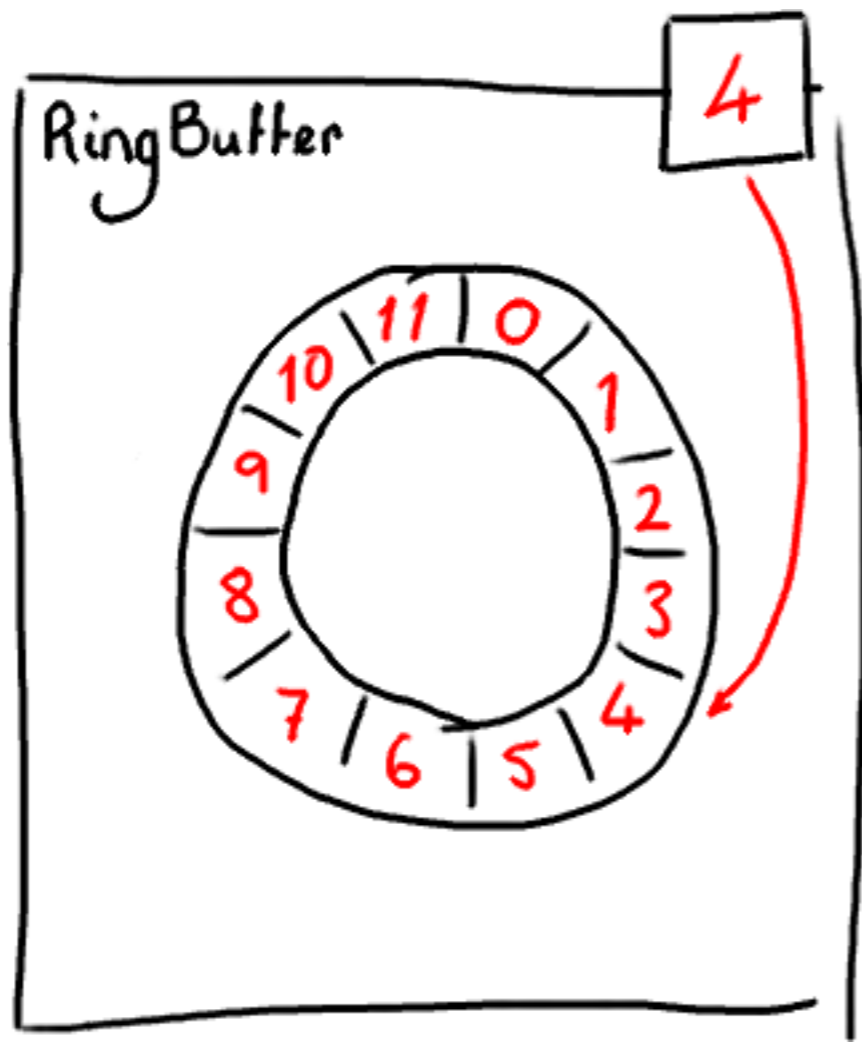


# ...so now I want to read

- The Disruptor provides nice batching behaviour for free

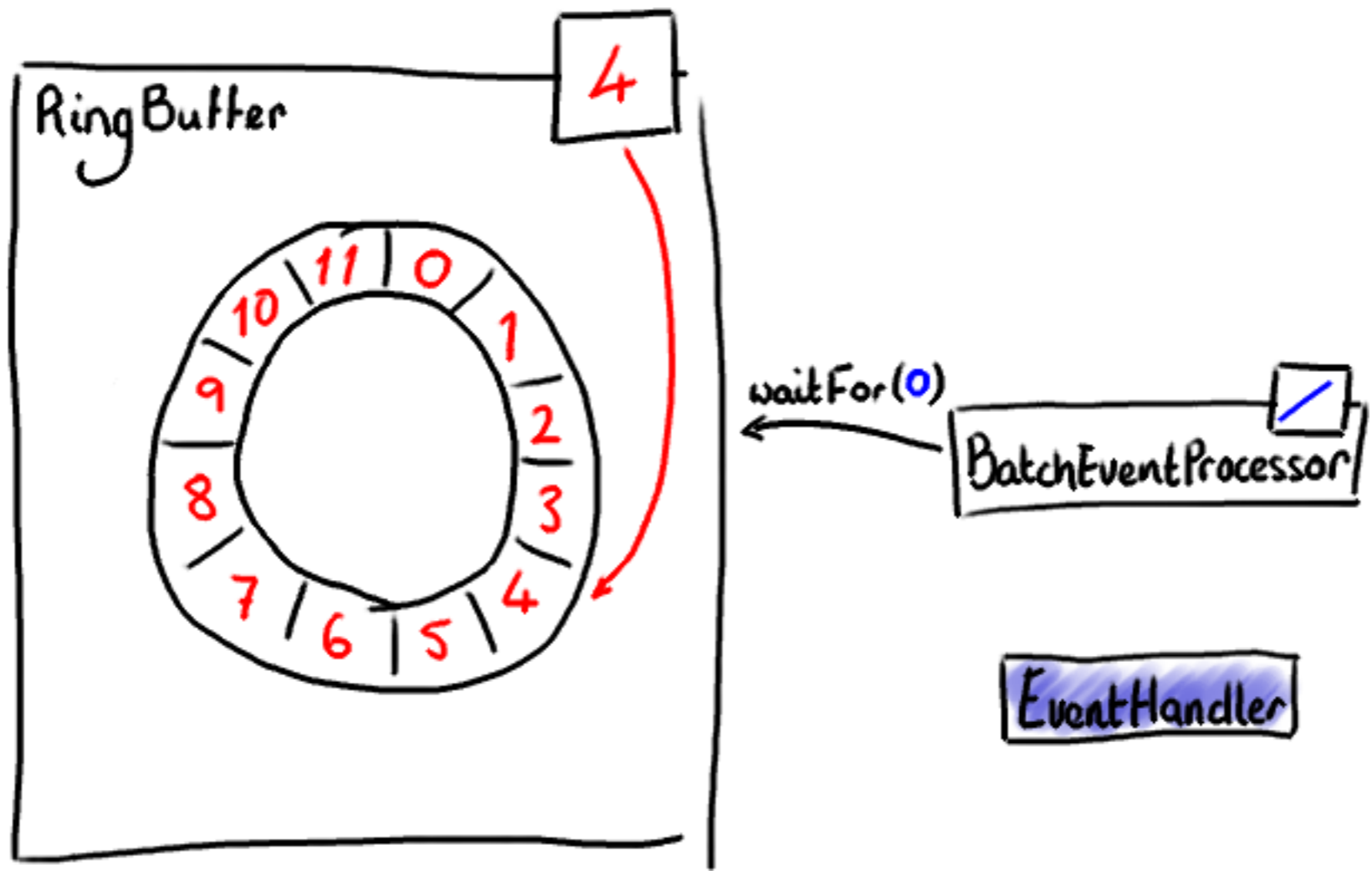


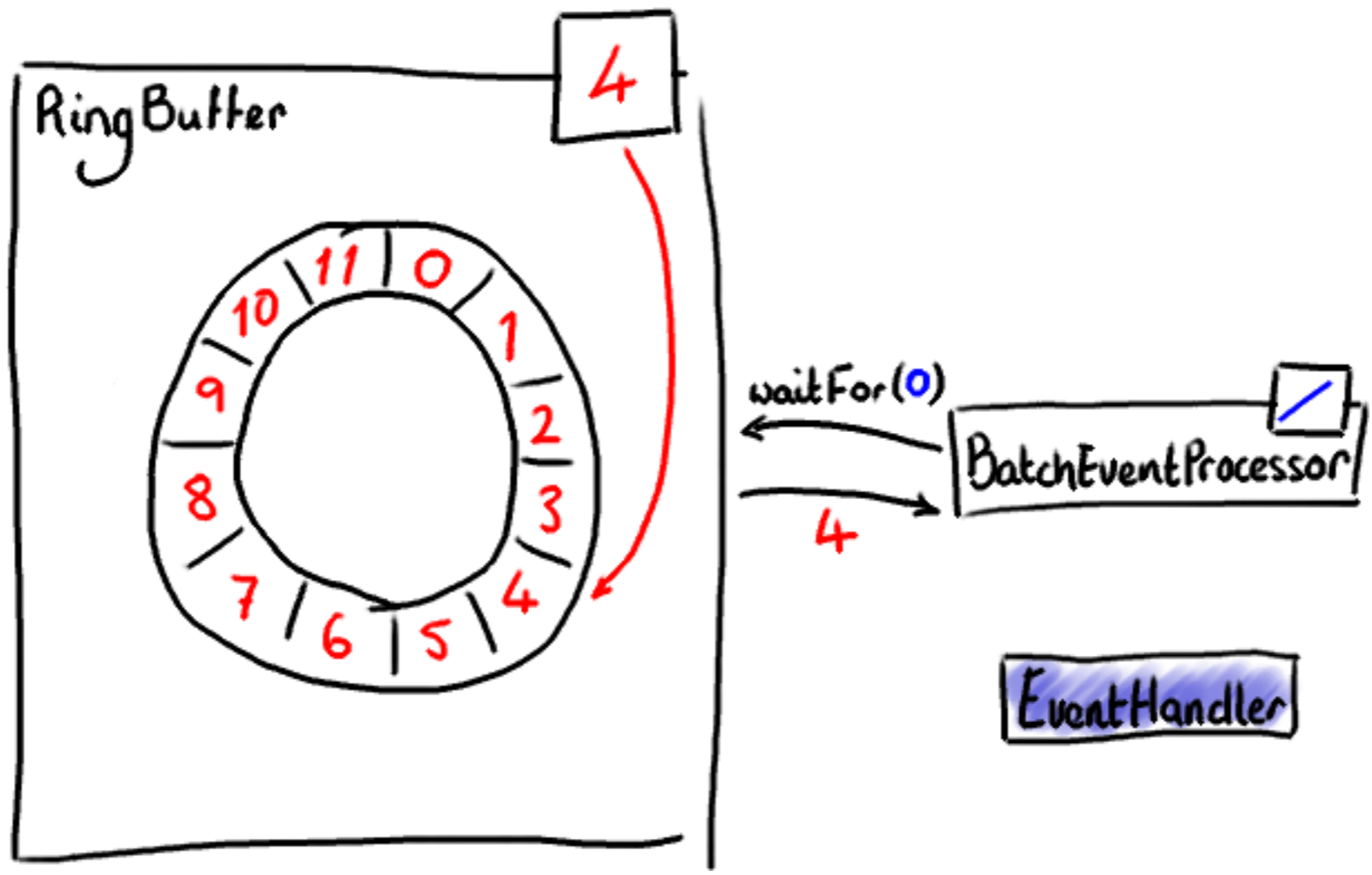


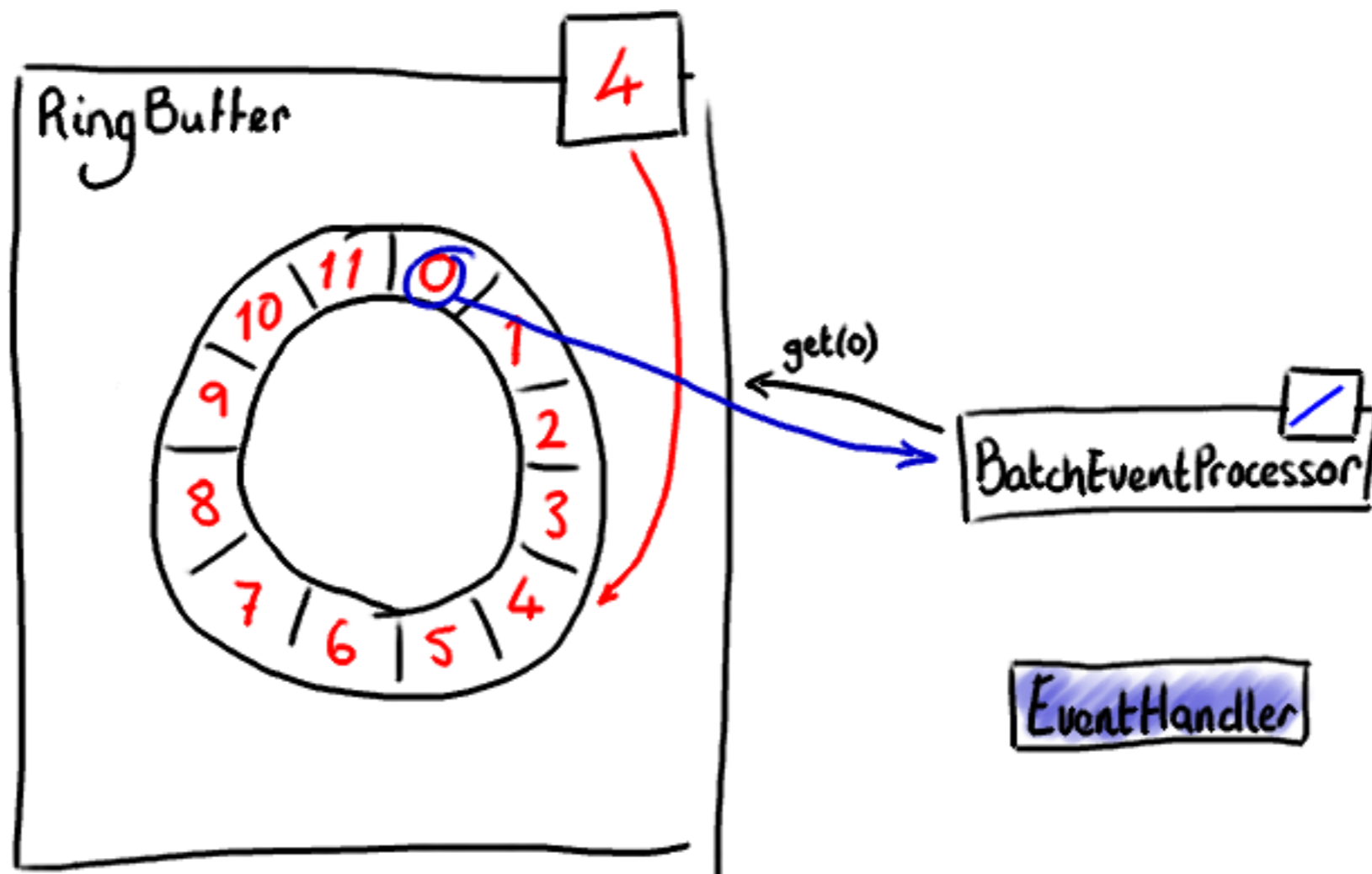


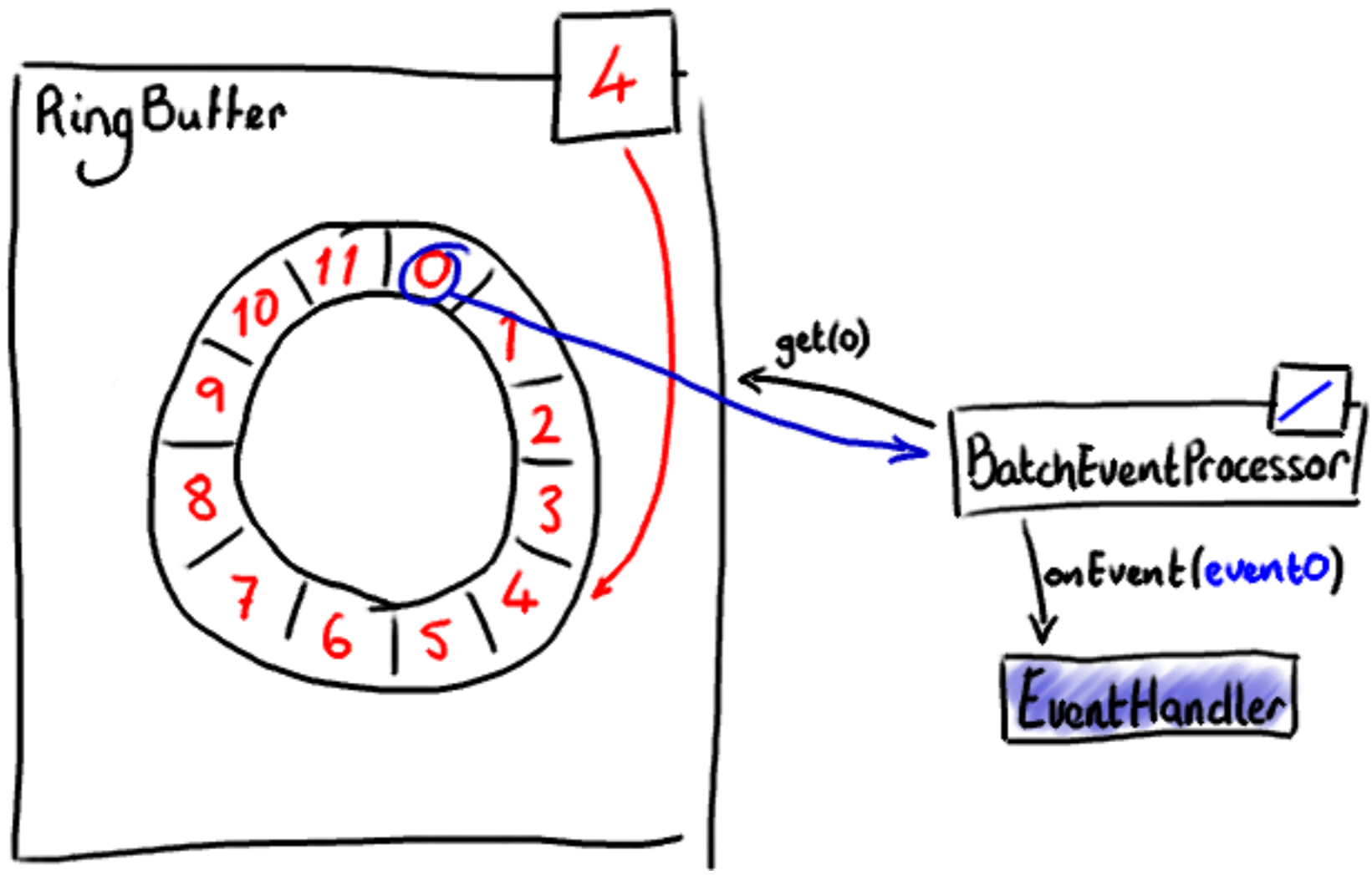
BatchEventProcessor

EventHandler

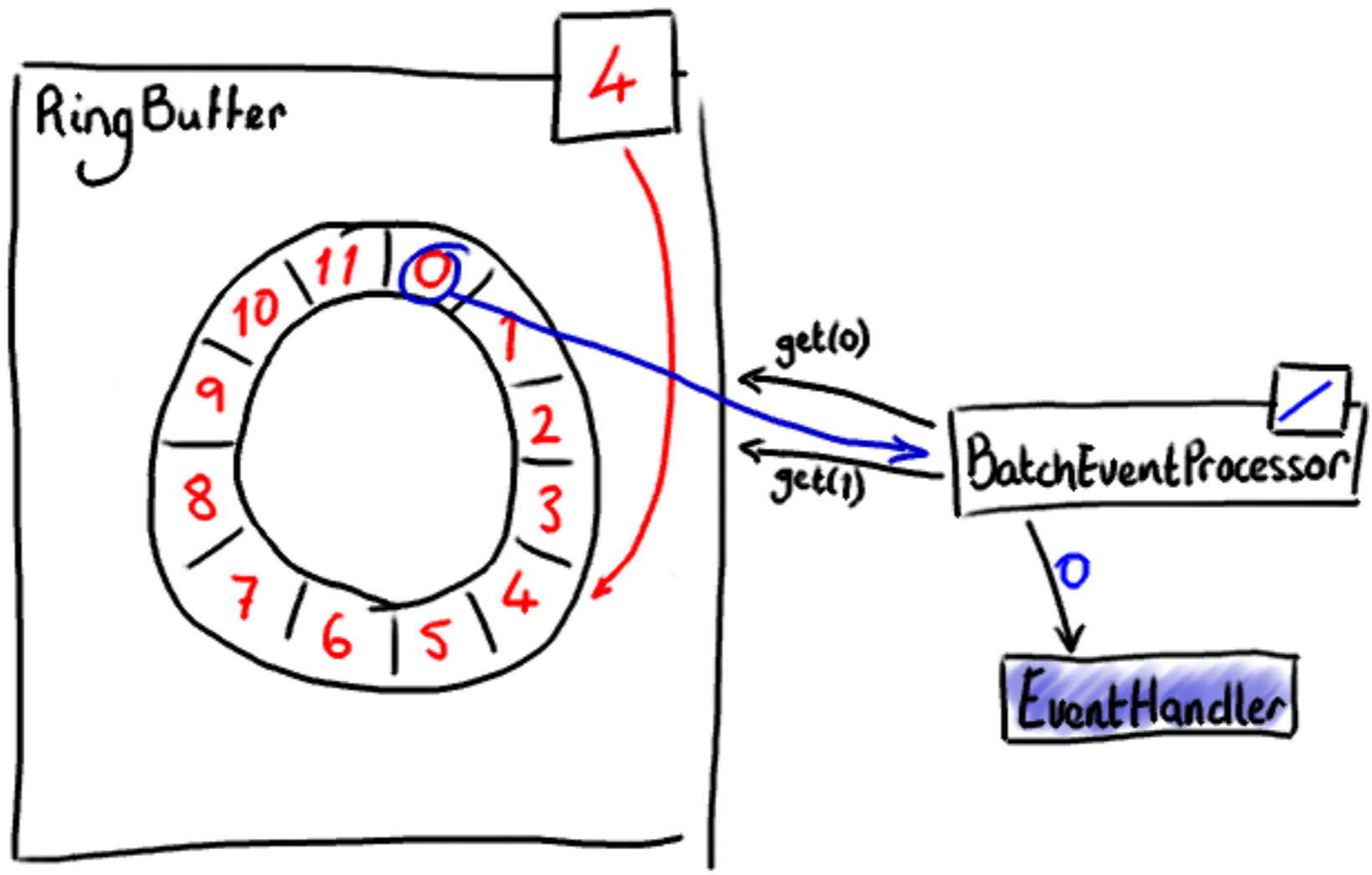


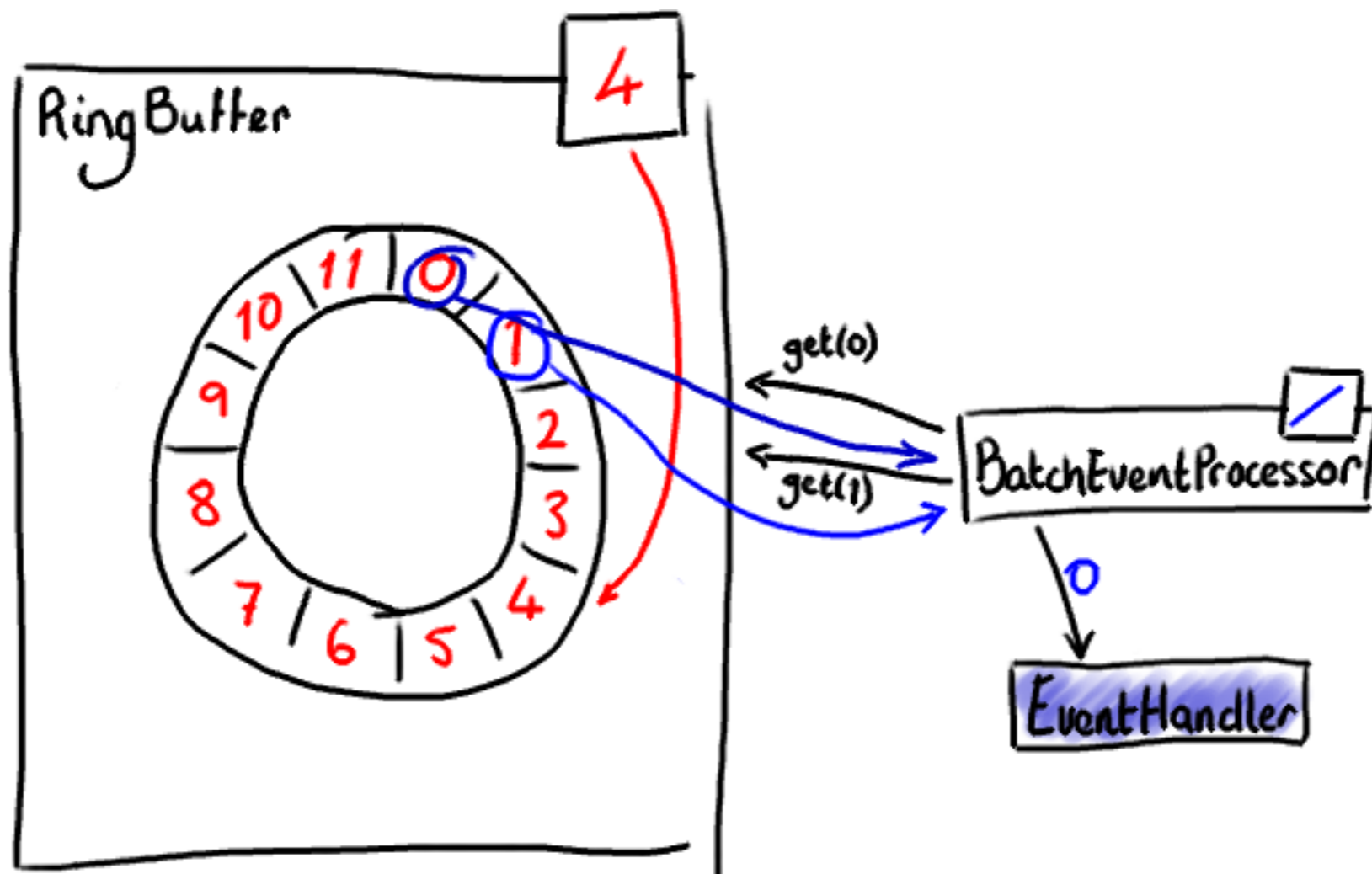


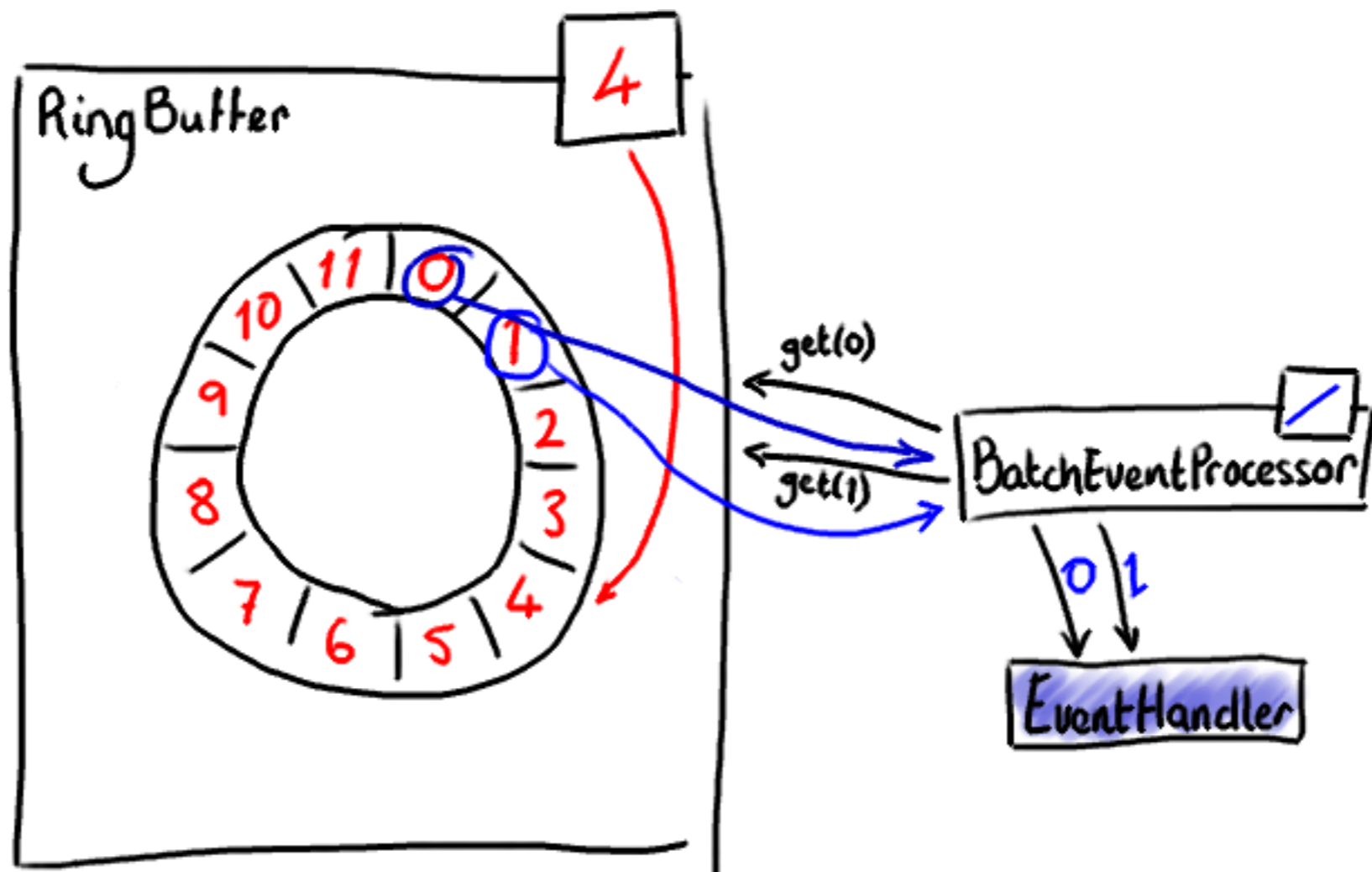


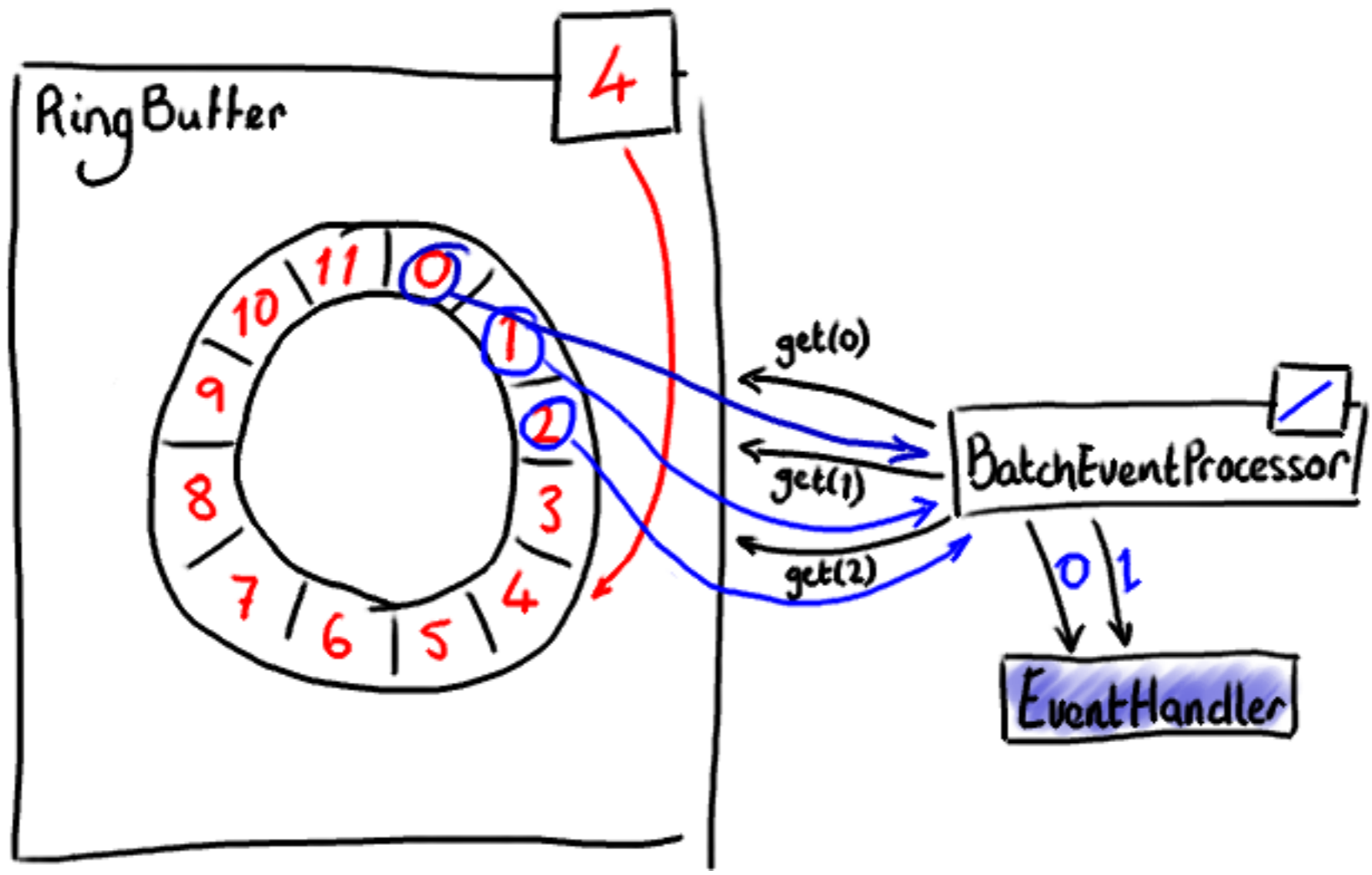


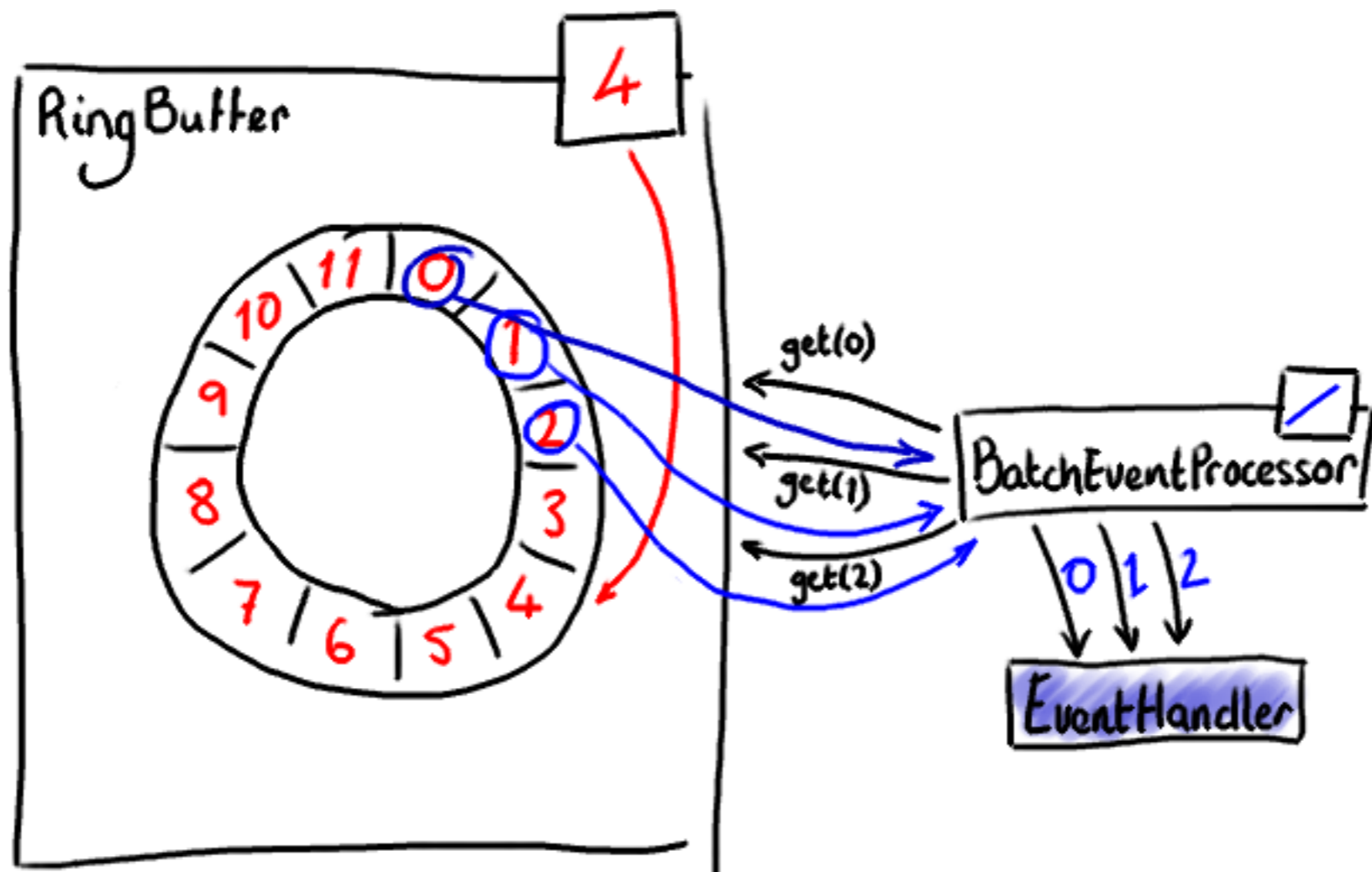


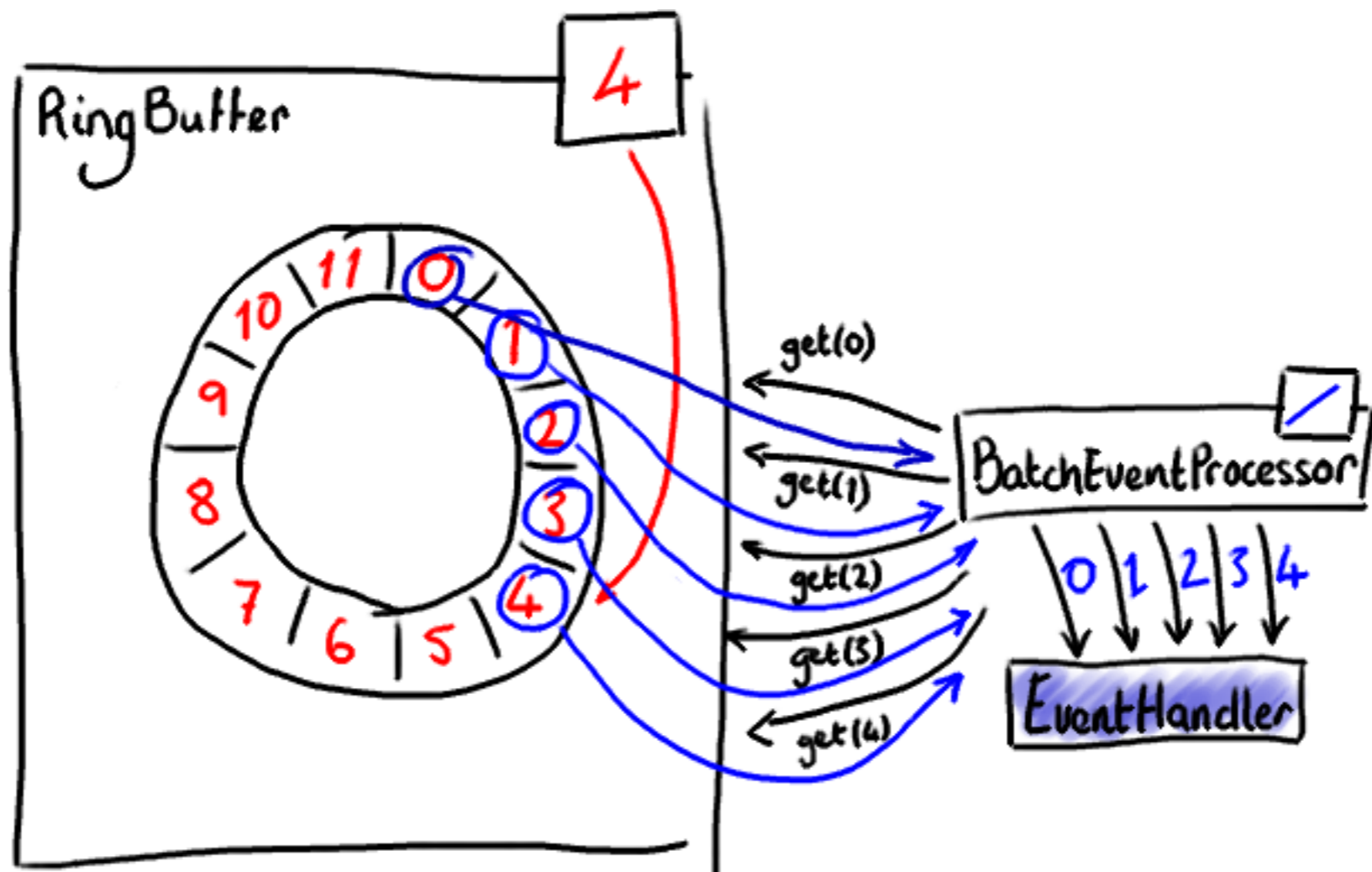


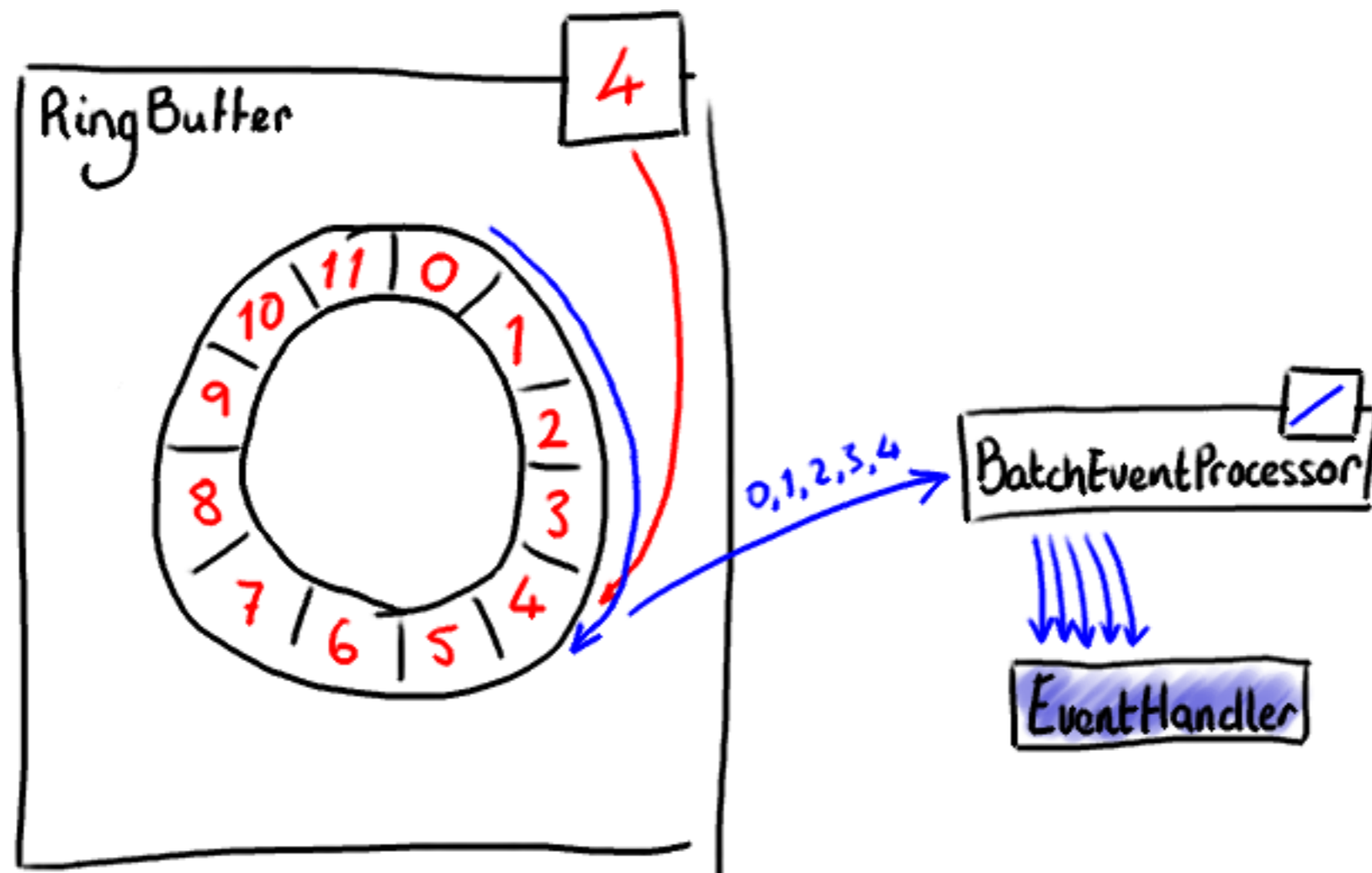


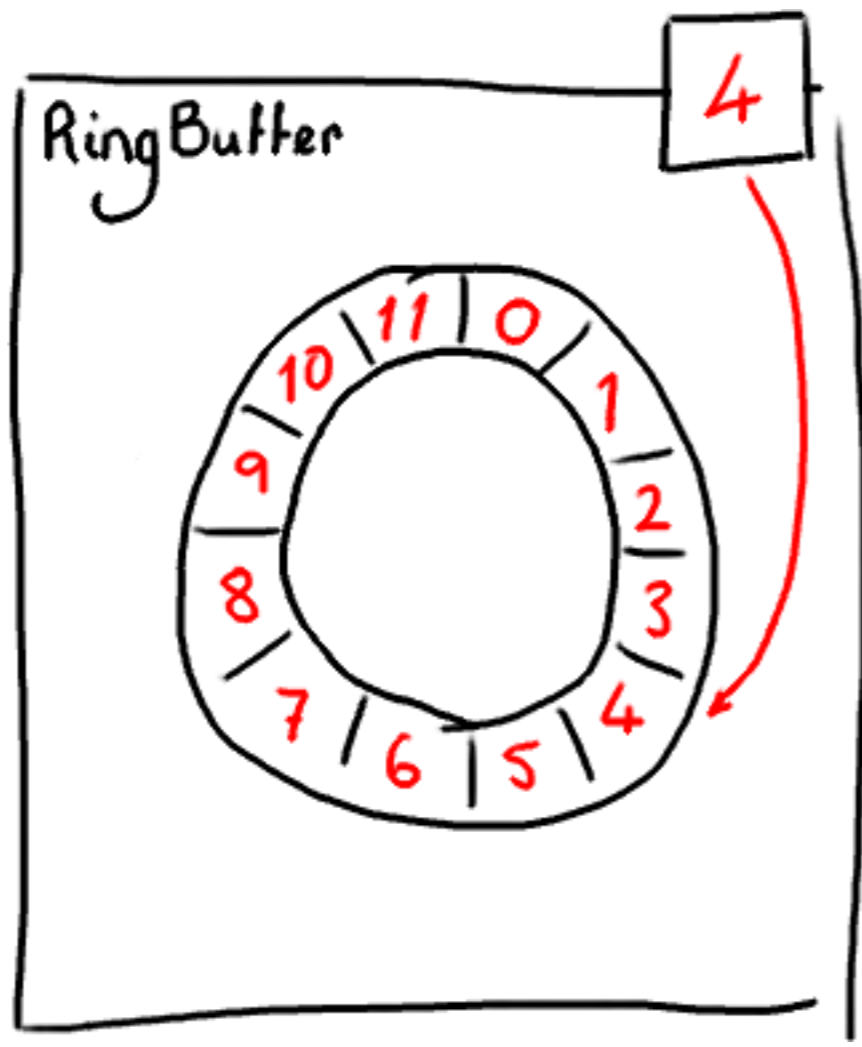












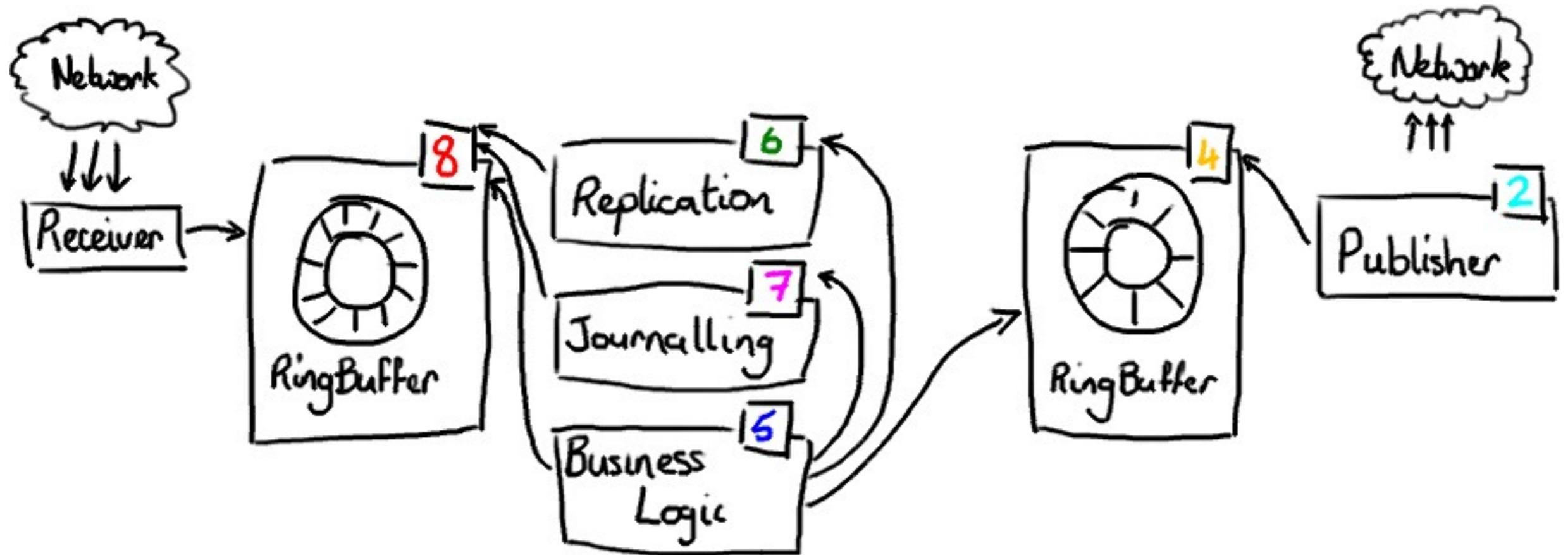


# ...and all you need is...

```
public class SimpleEventHandler implements EventHandler<SimpleEvent>
{
    @Override
    public void onEvent(final SimpleEvent event,
                       final long sequence,
                       final boolean endOfBatch) throws Exception {
        // do stuff
    }
}
```

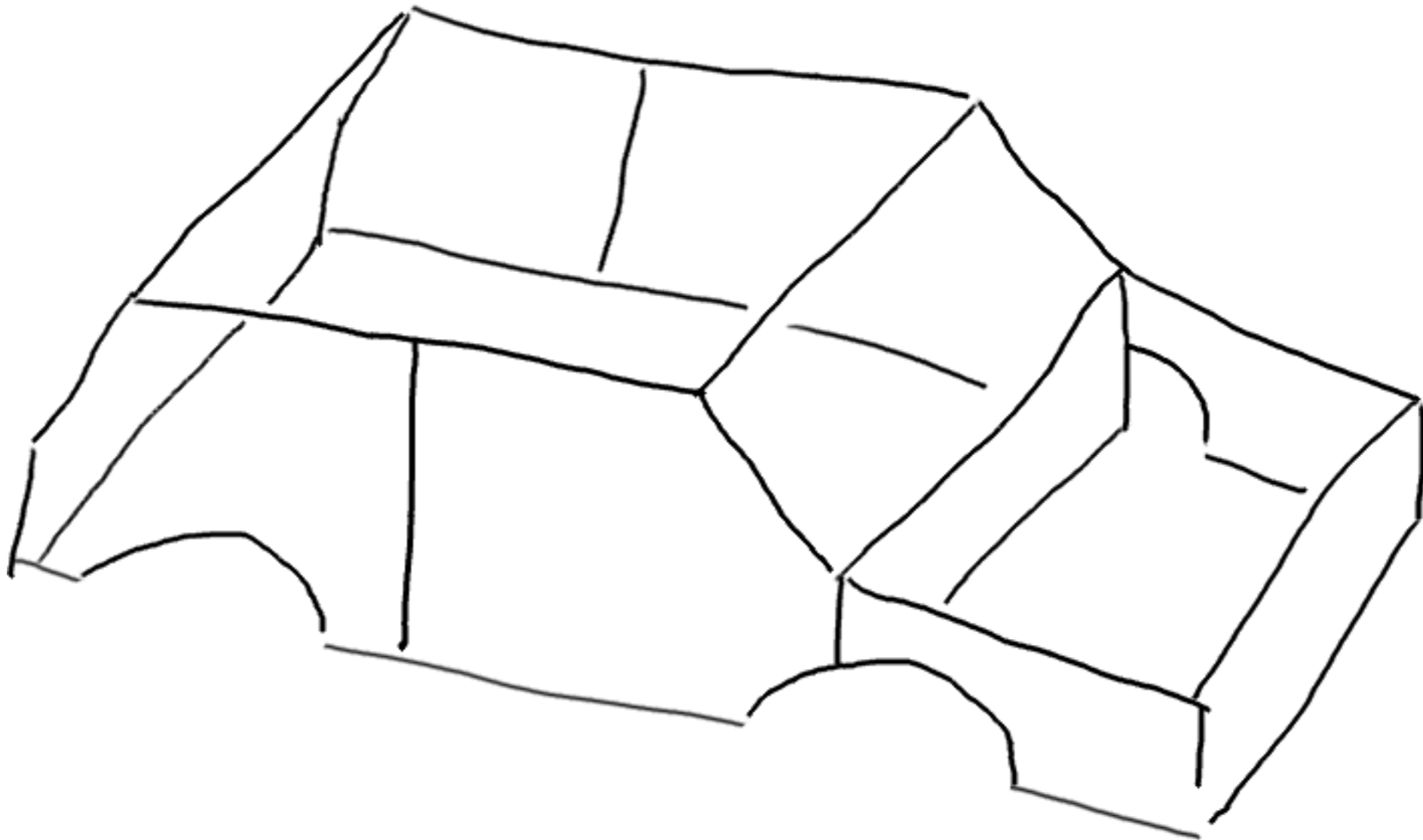
**Shiny. So what?**

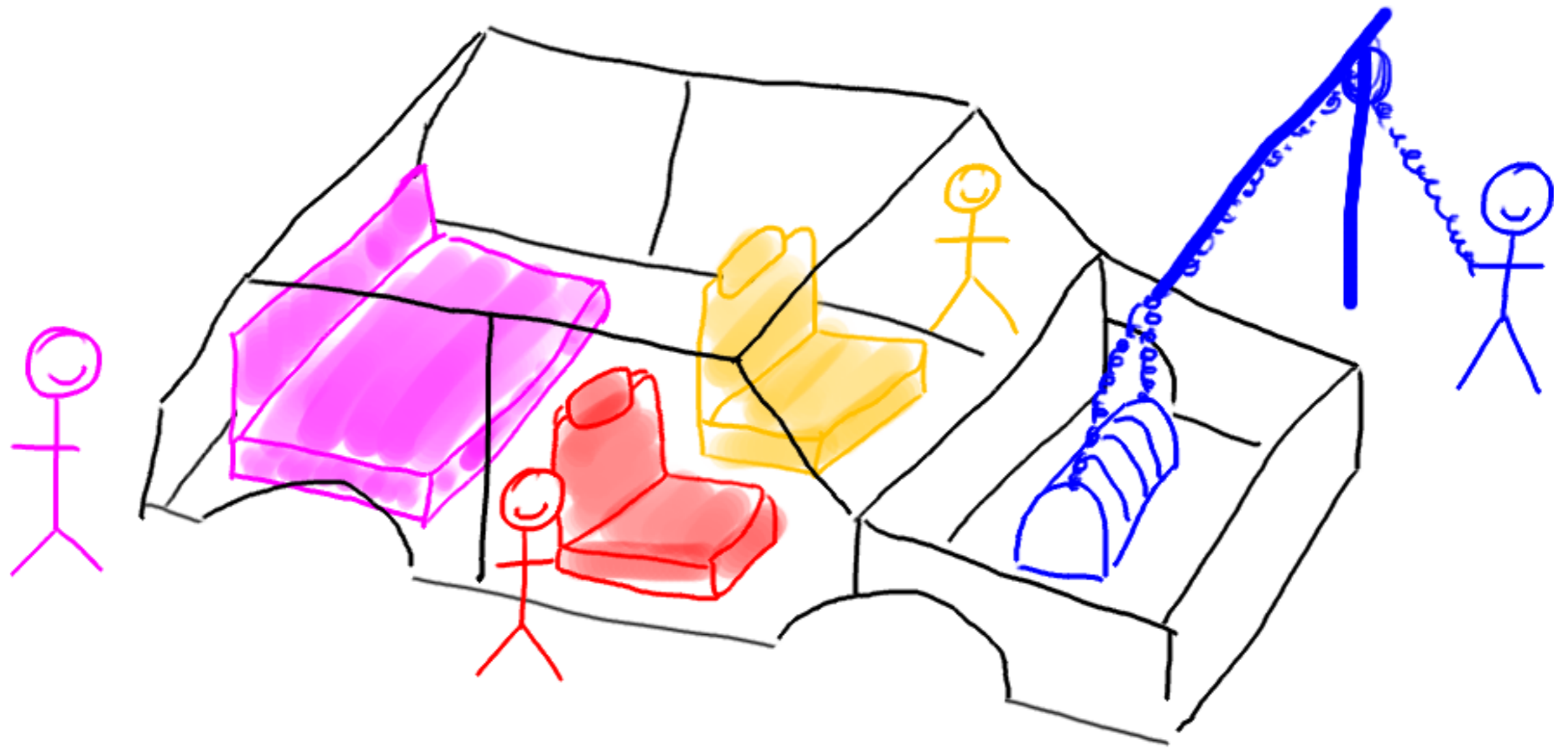
# Let's go parallel



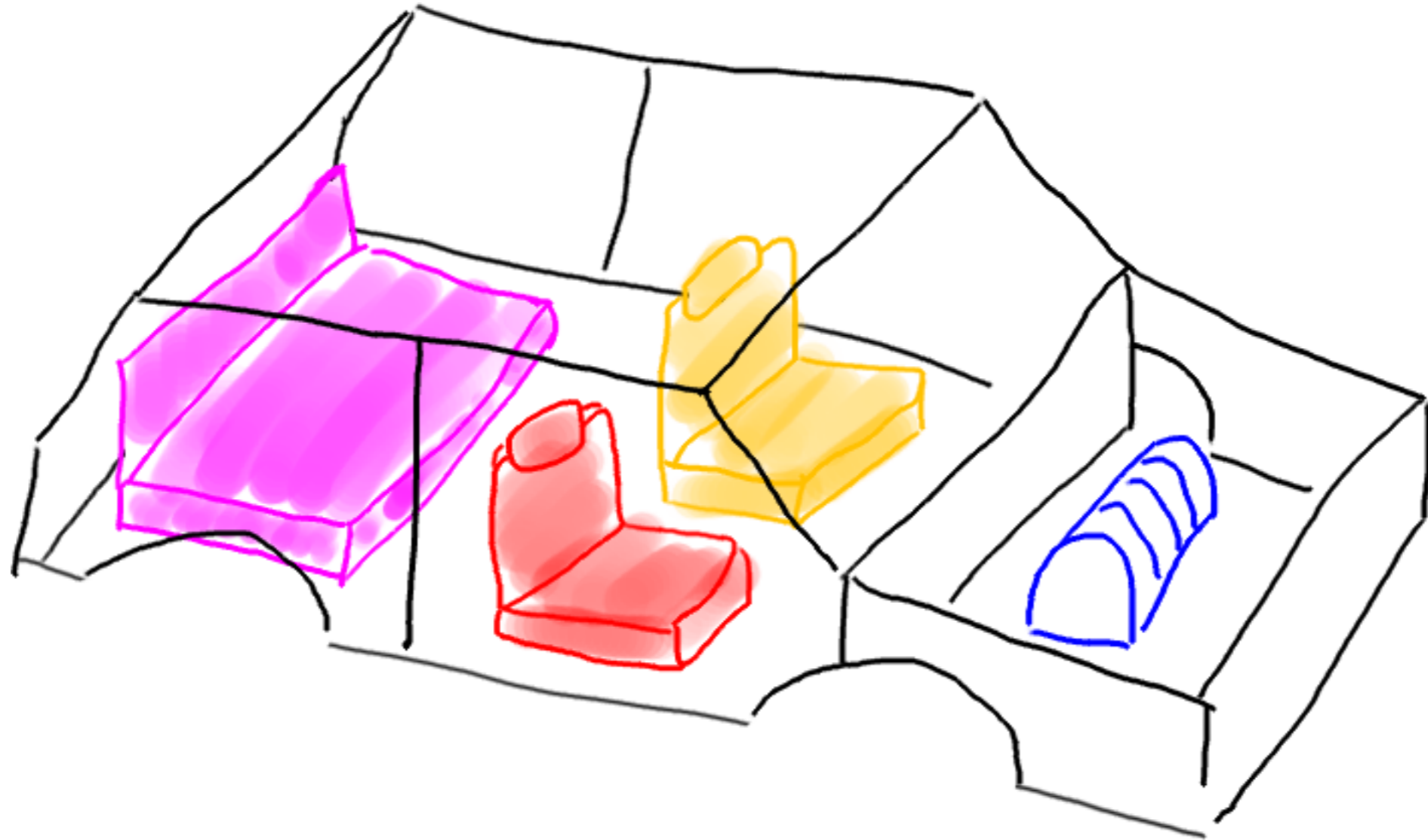
**And now for something  
different...**

# Remember Henry Ford?

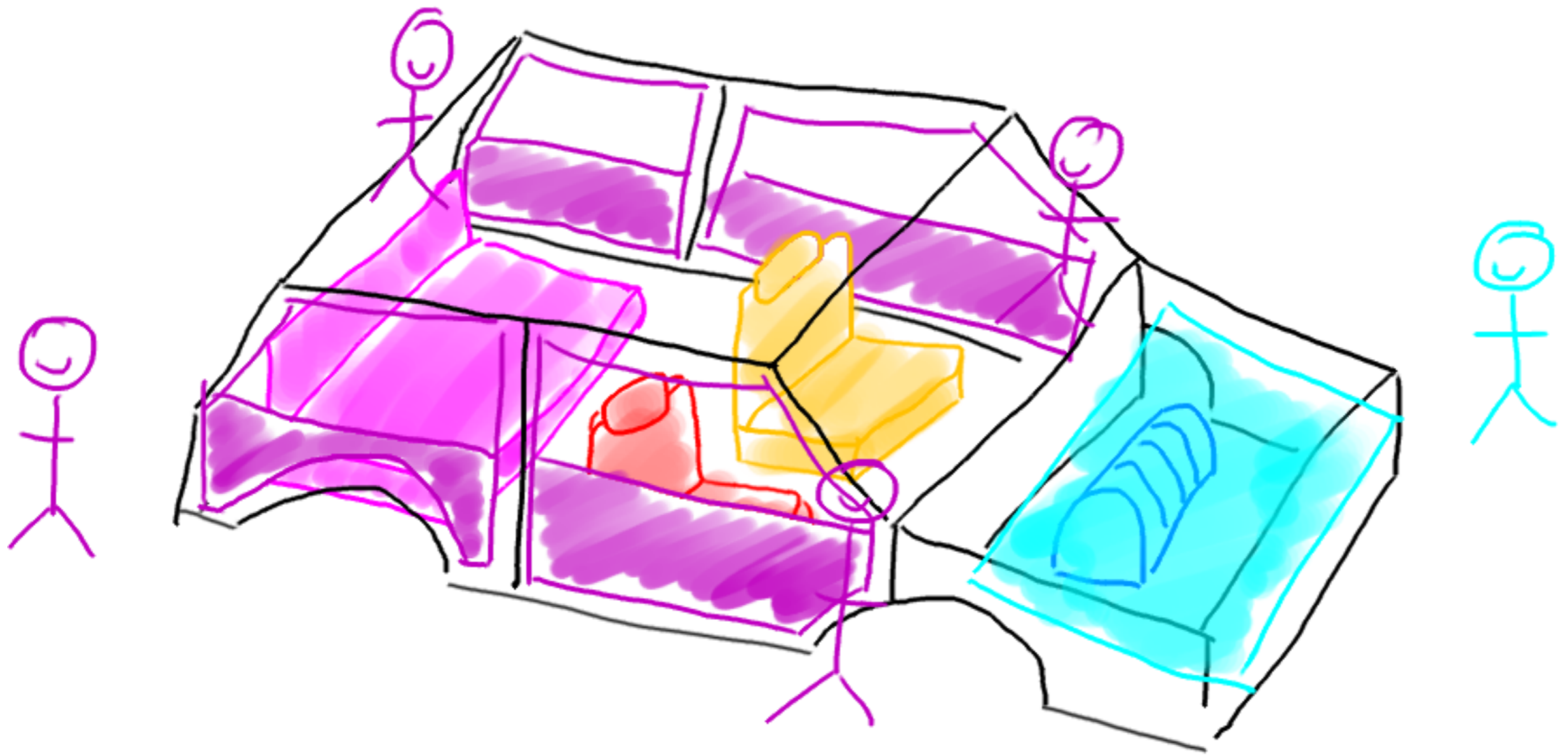




\*Not to Scale

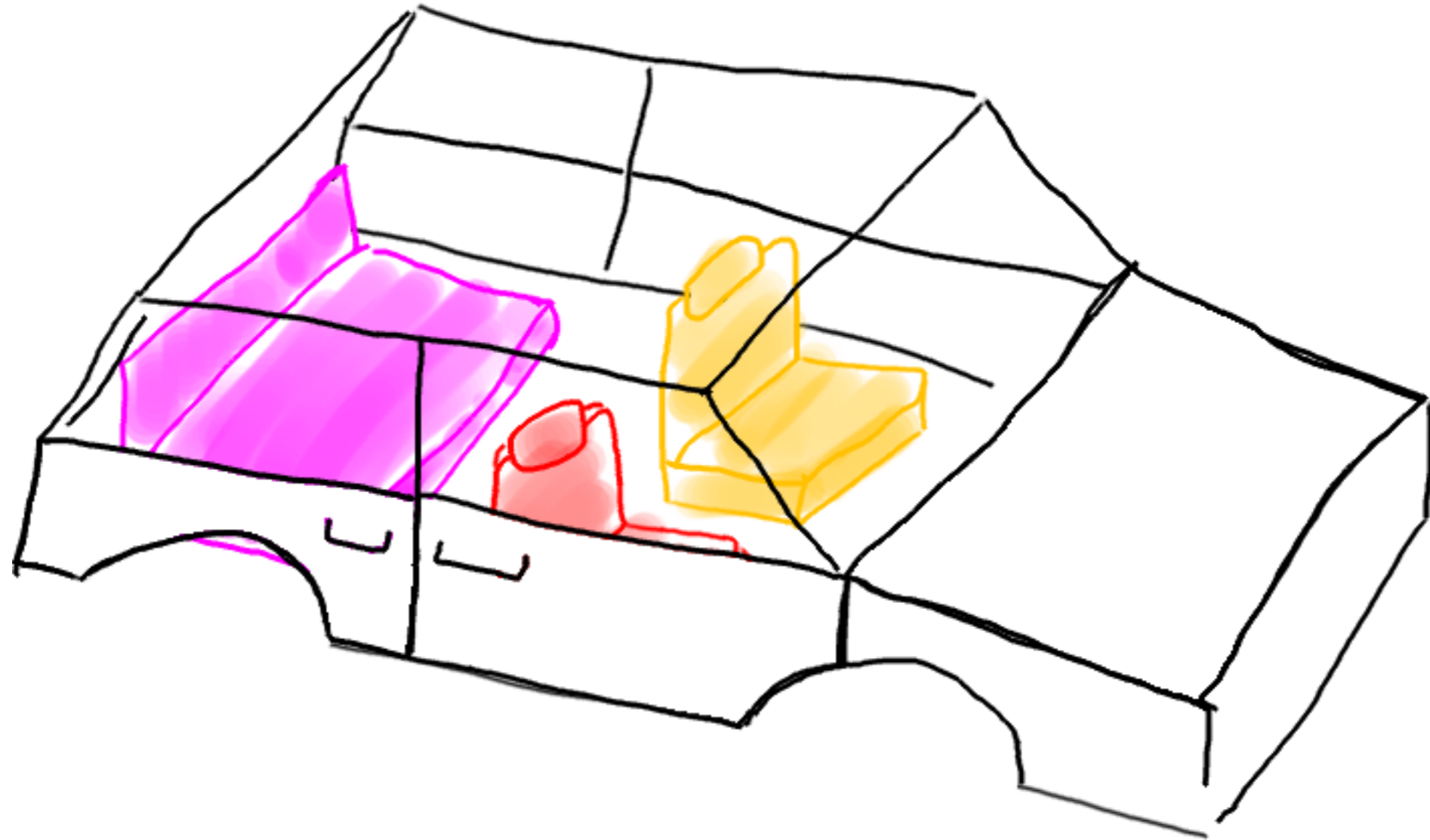


\*Not to Scale

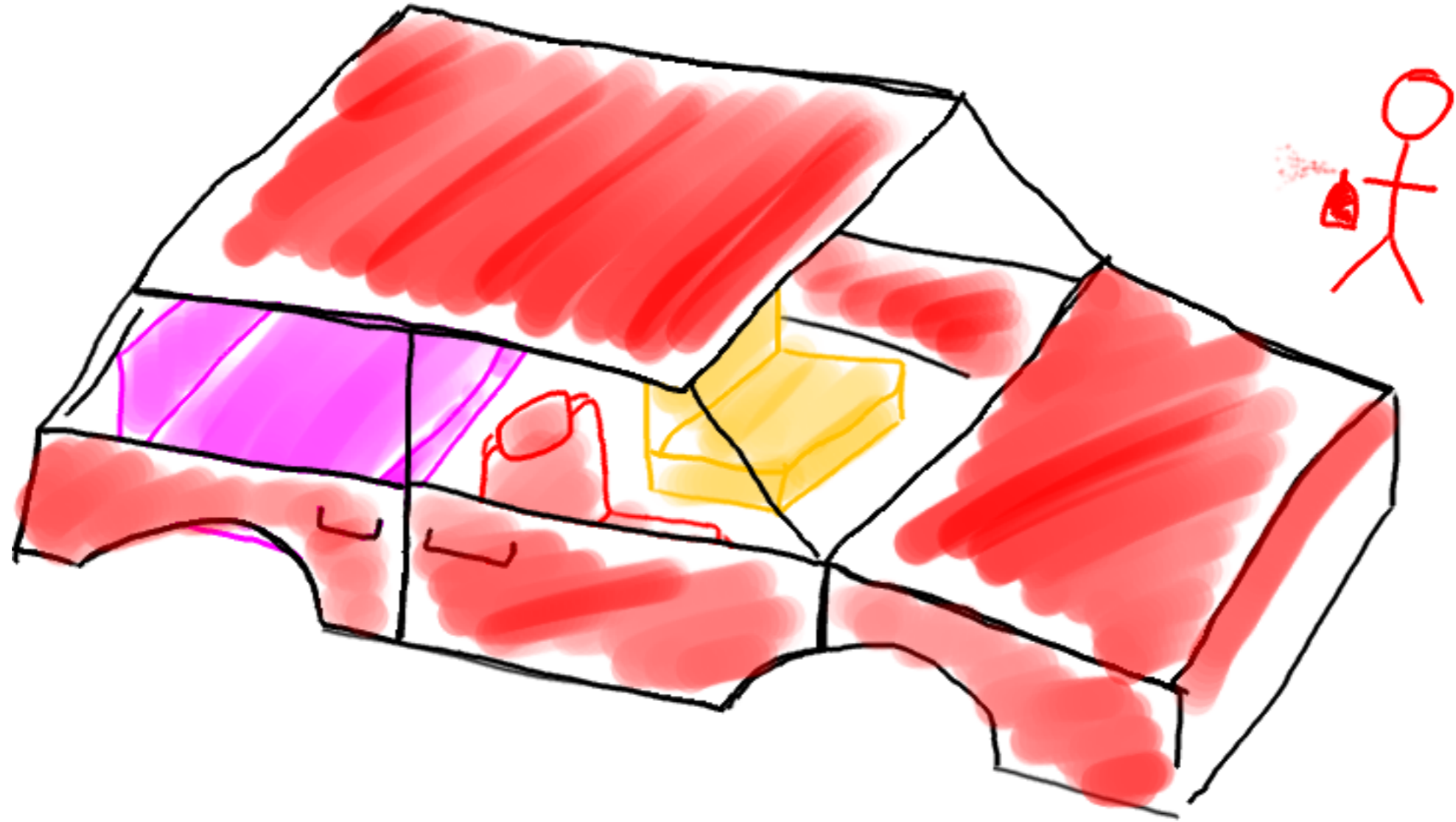


\*Not to Scale

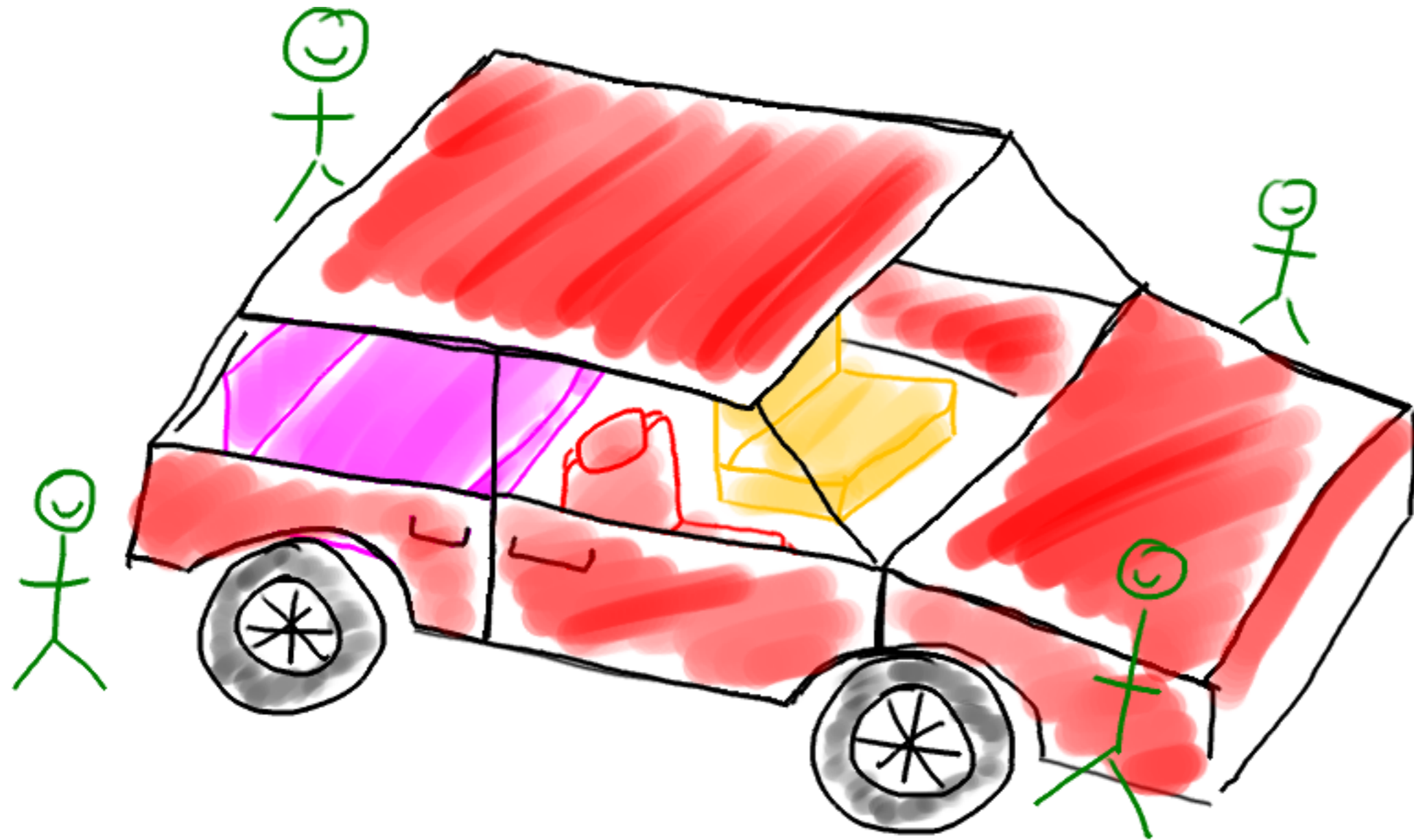




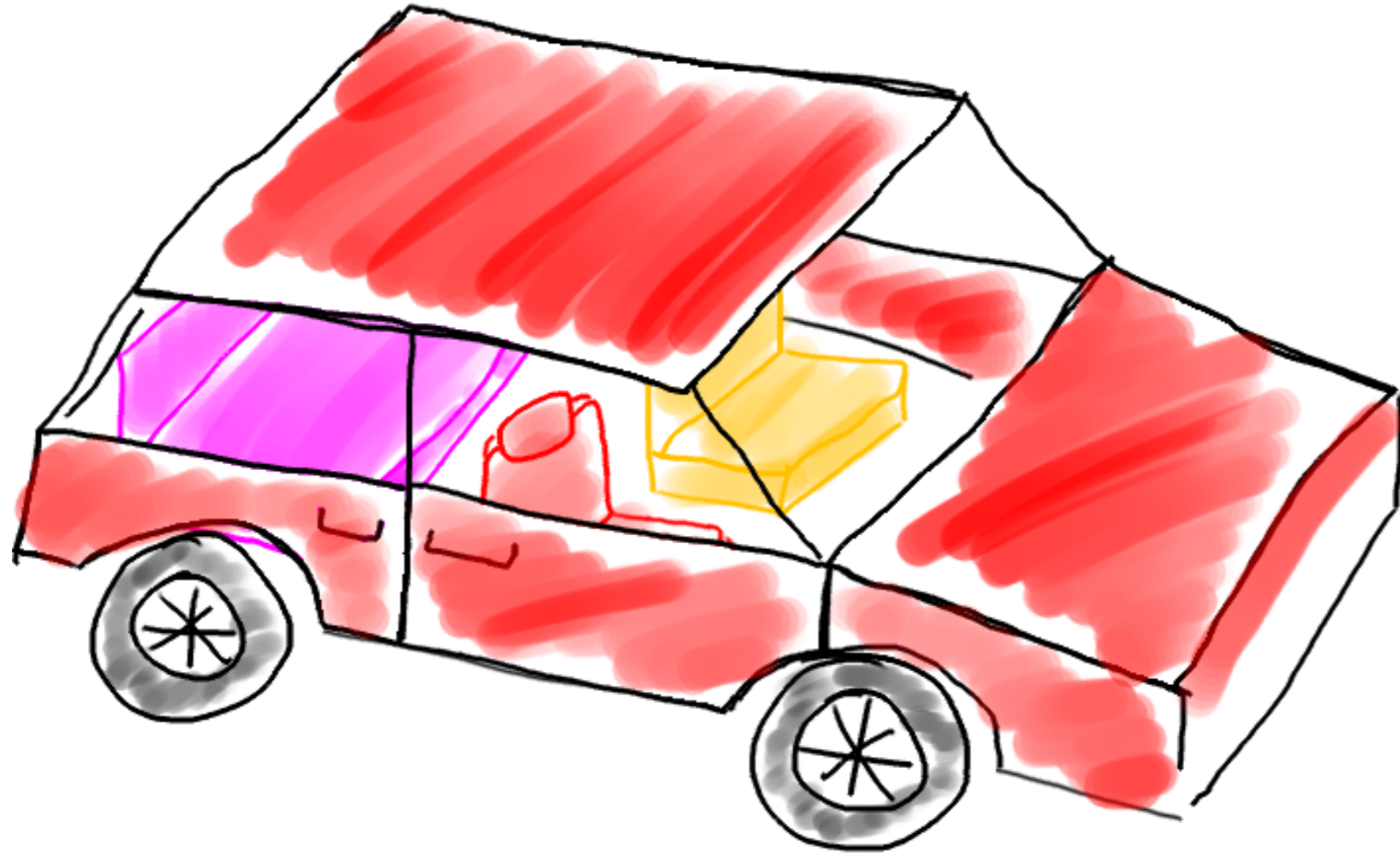
\*Not to Scale



\*Not to Scale

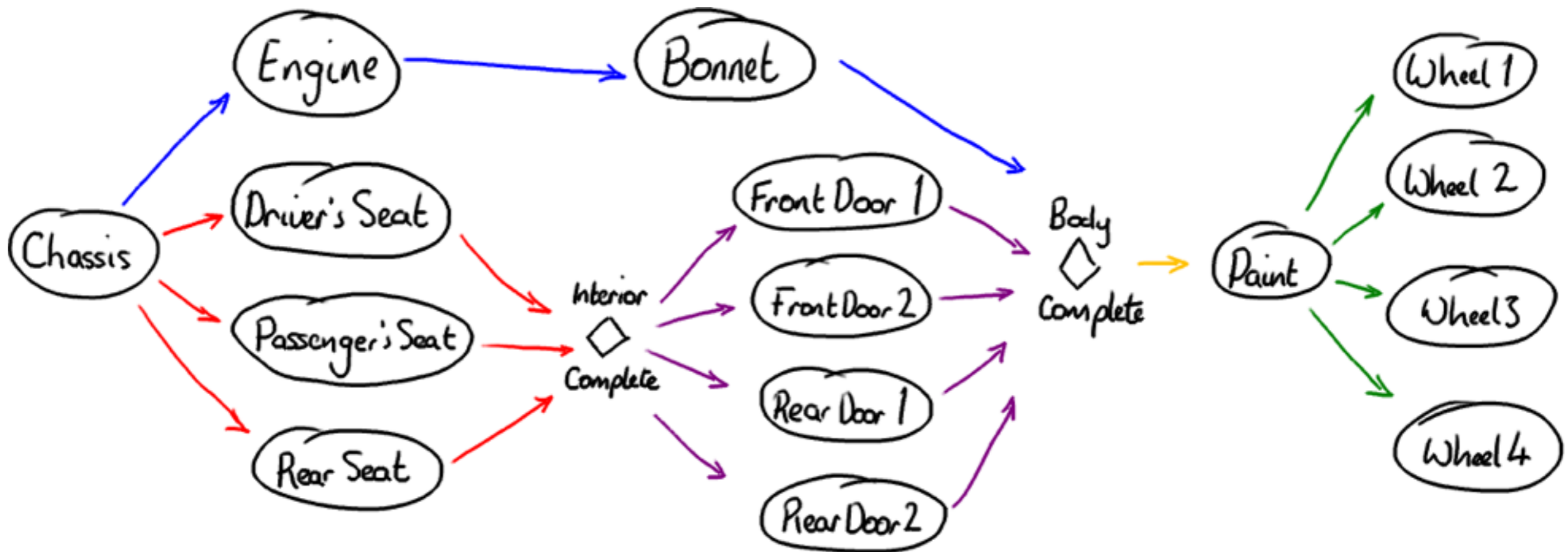


\*Not to Scale



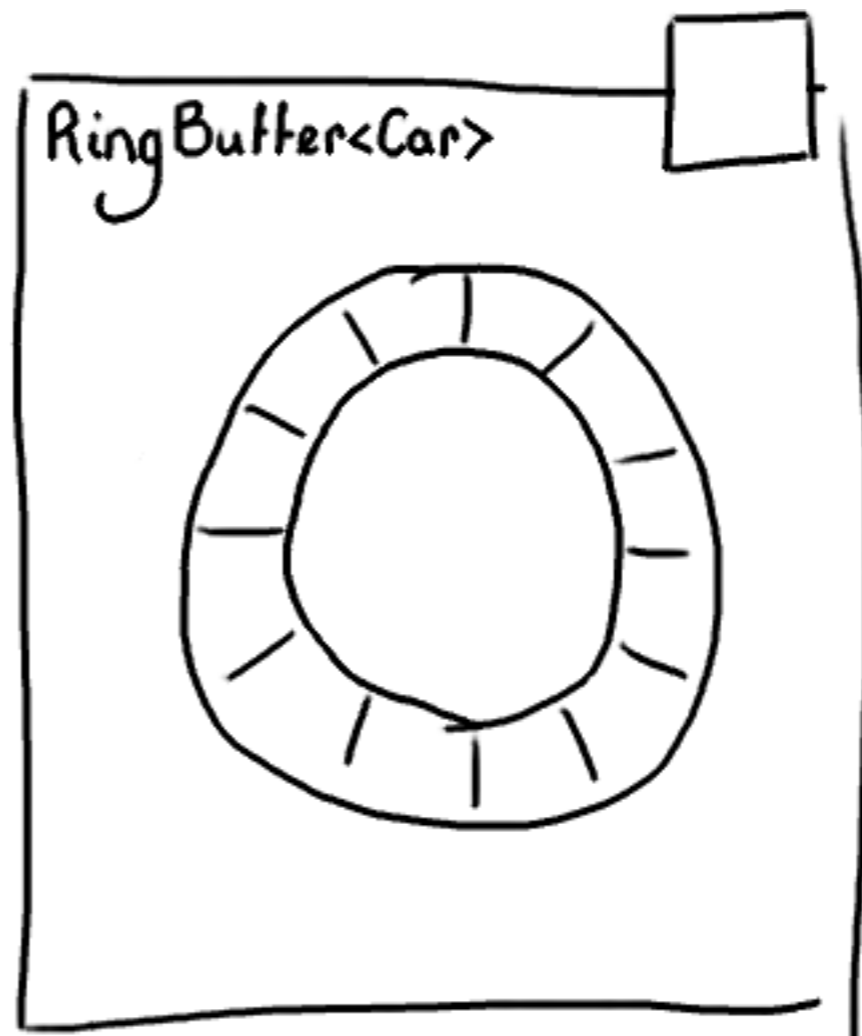
\*Not to Scale

# Complex workflow...

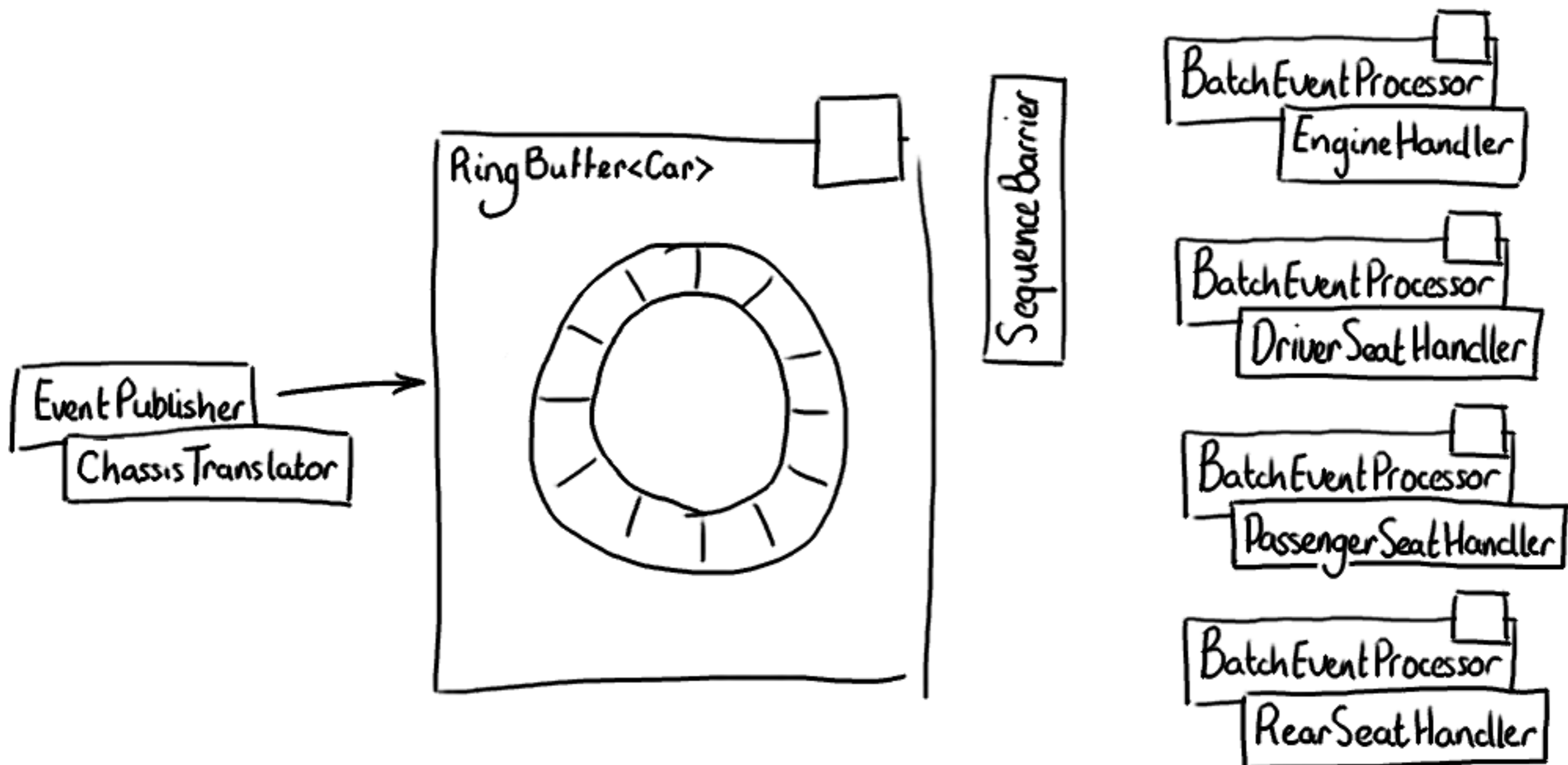


**What on Earth has this  
got to do with  
RingBuffers?!**

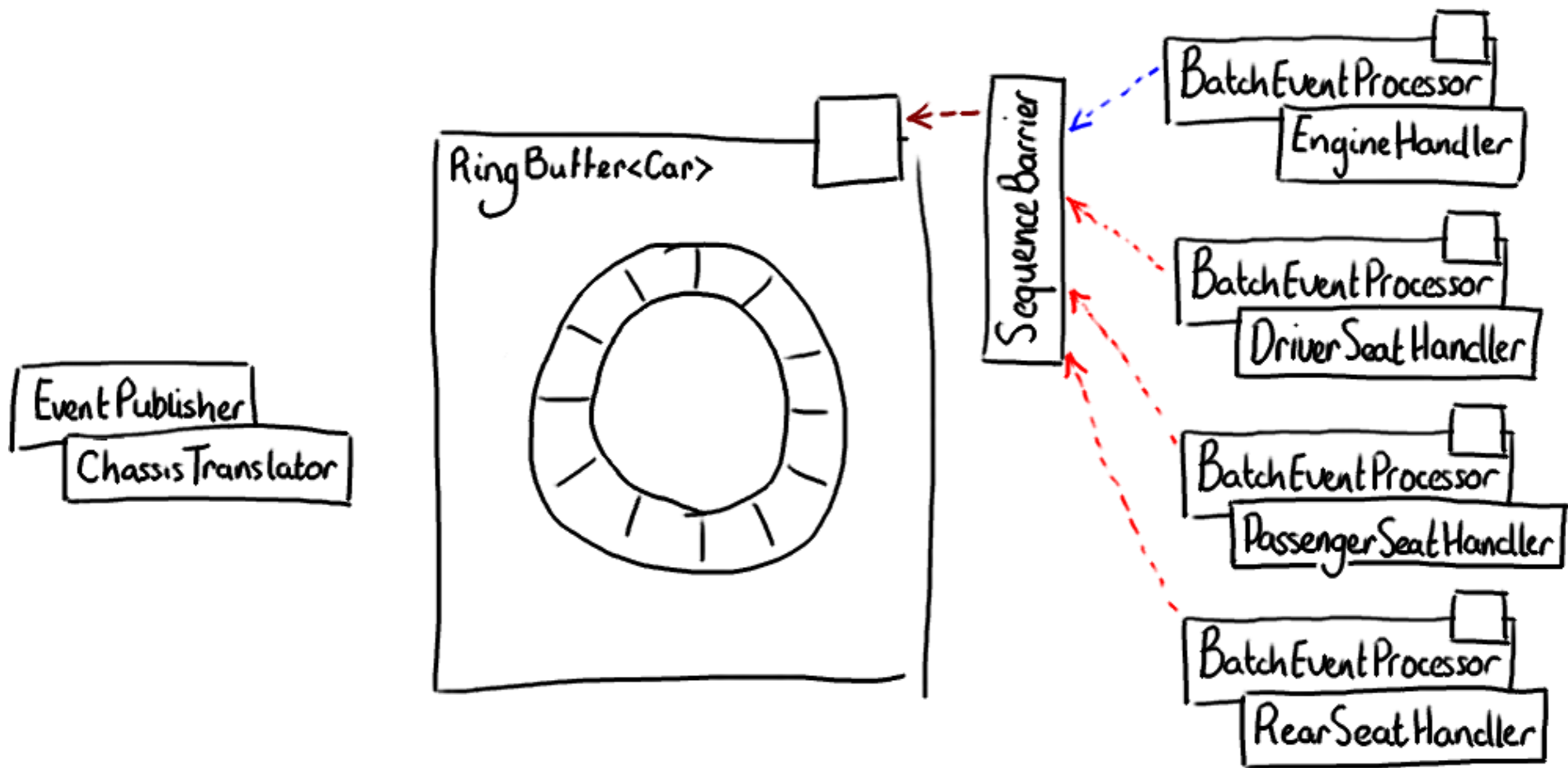
Event Publisher  
Chassis Translator

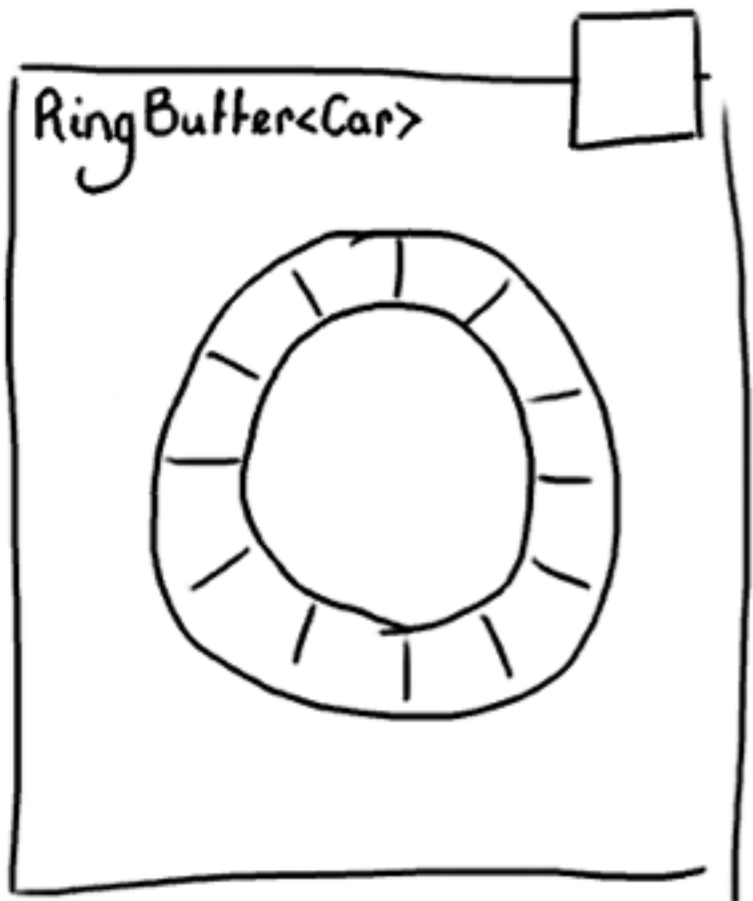




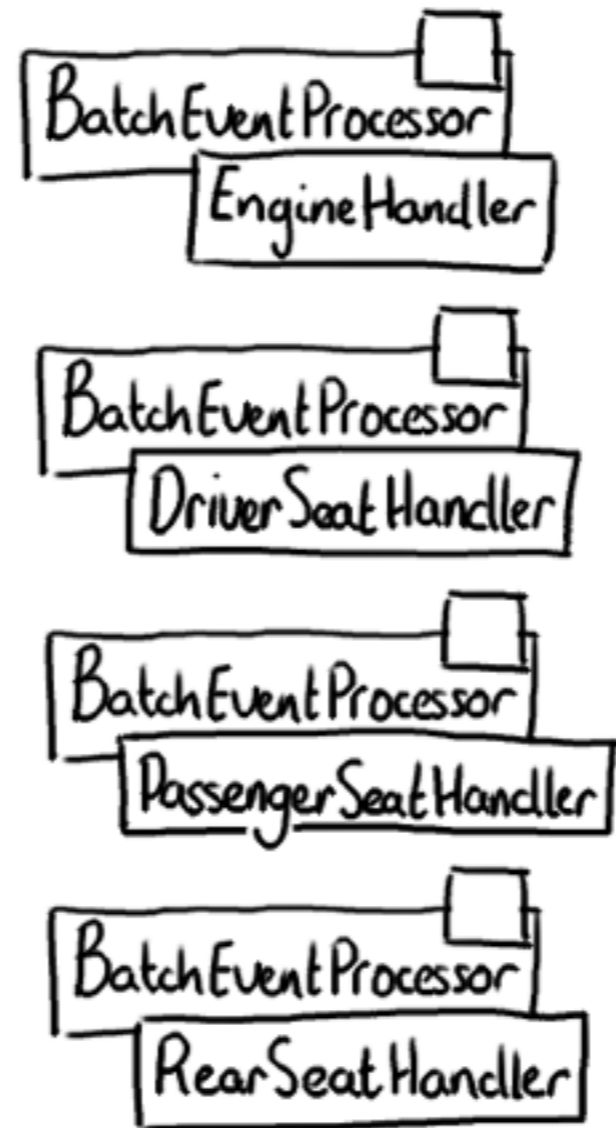






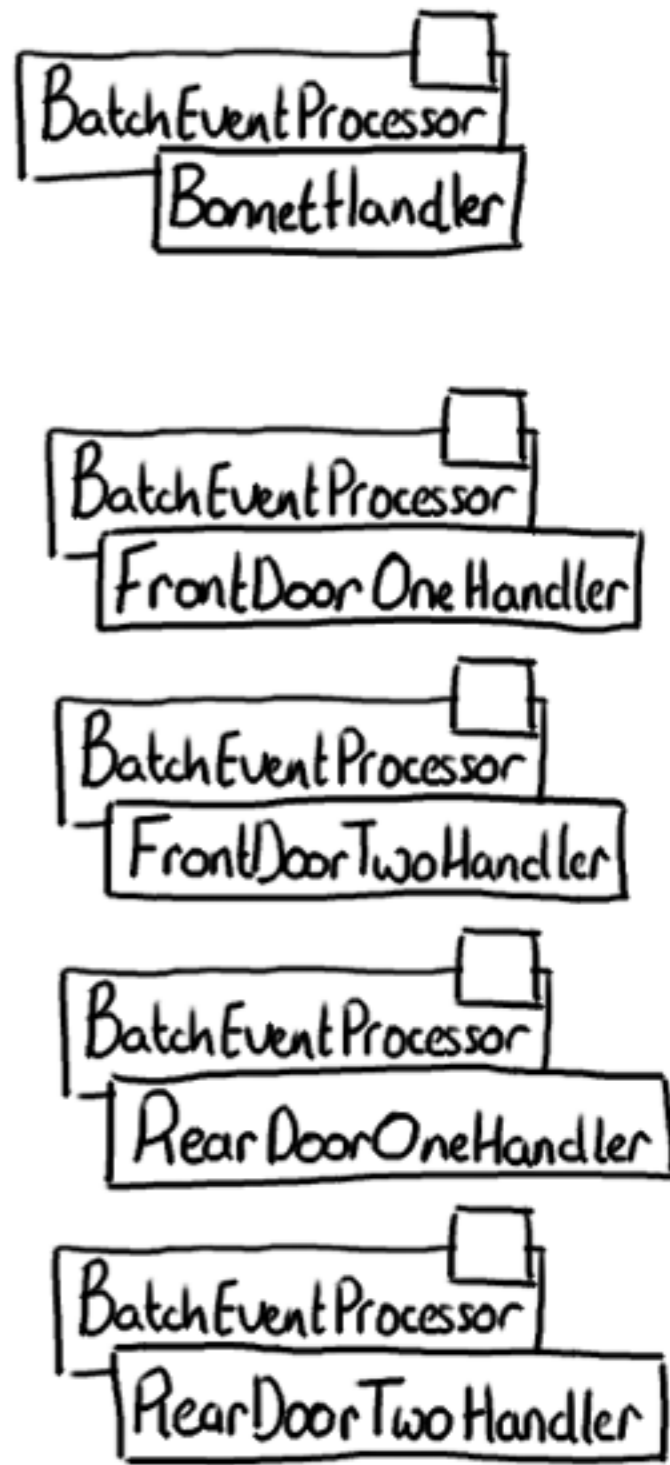


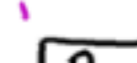
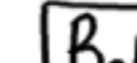
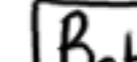
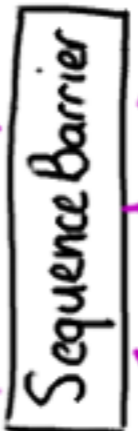
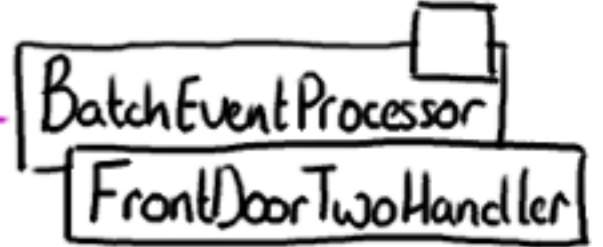
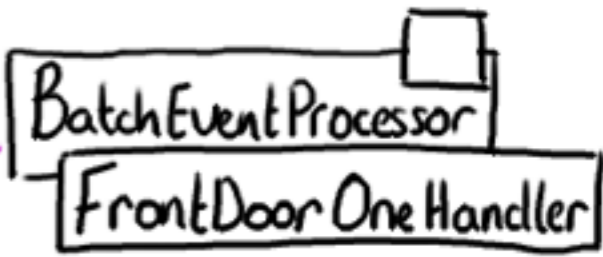
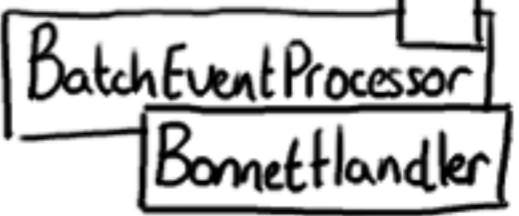
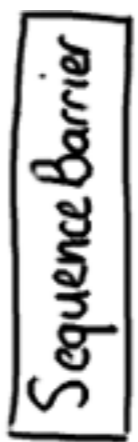
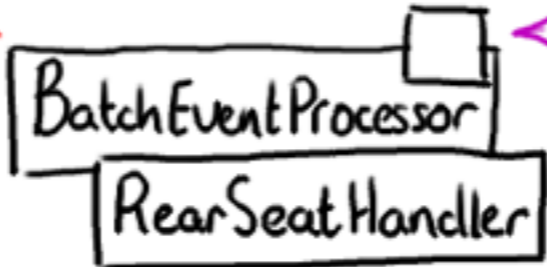
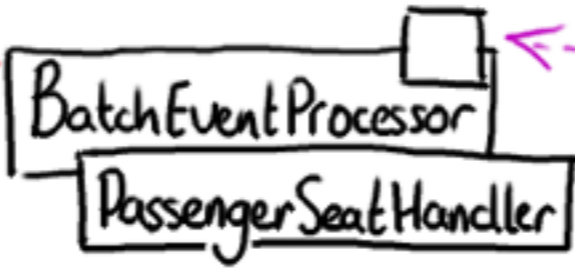
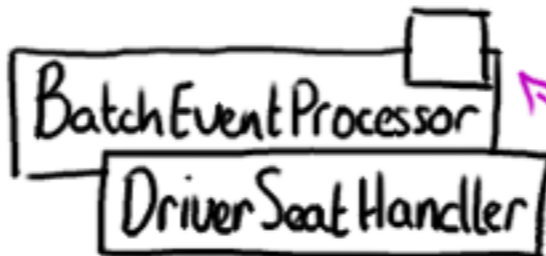
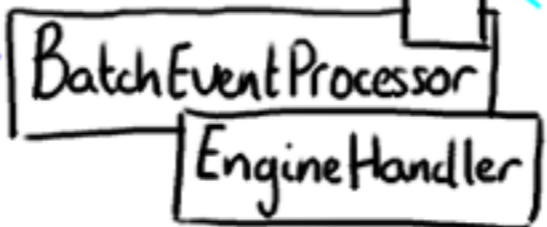
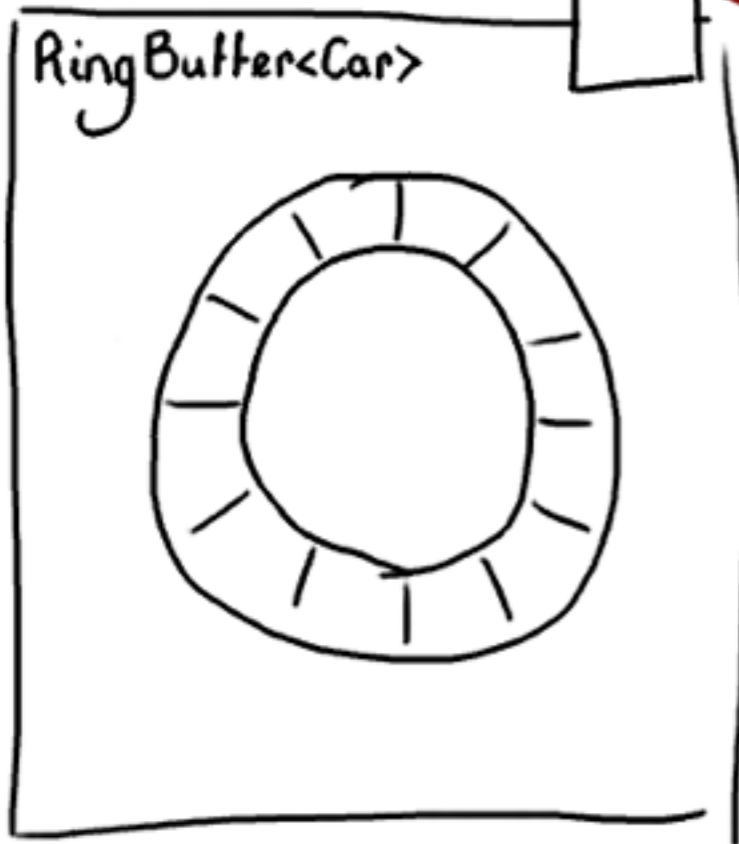
Sequence Barrier



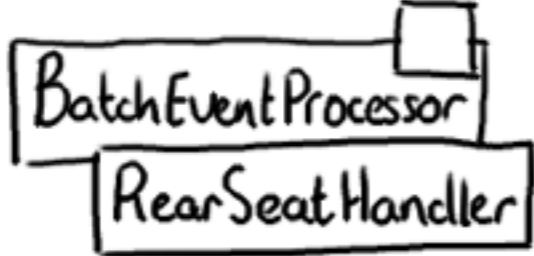
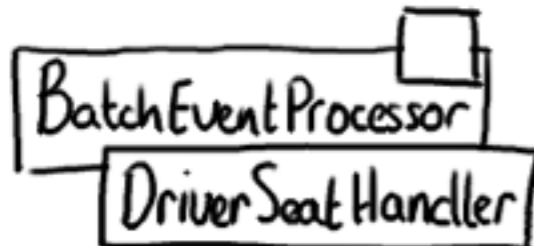
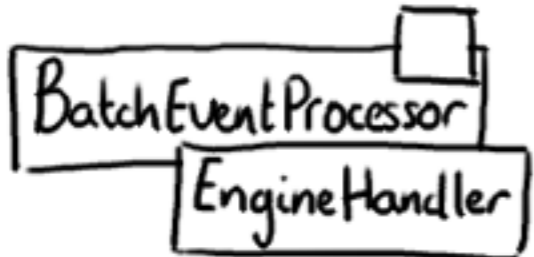
Sequence Barrier

Sequence Barrier



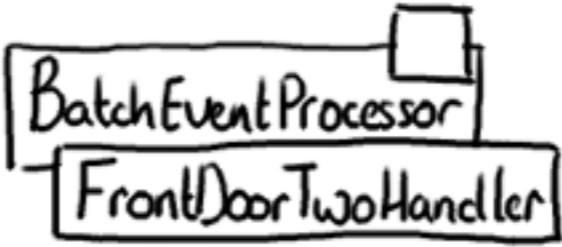
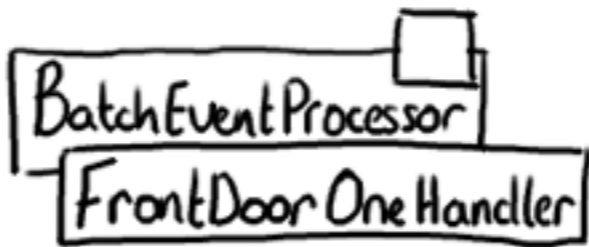
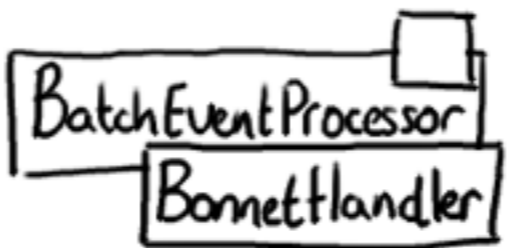


Sequence Barrier



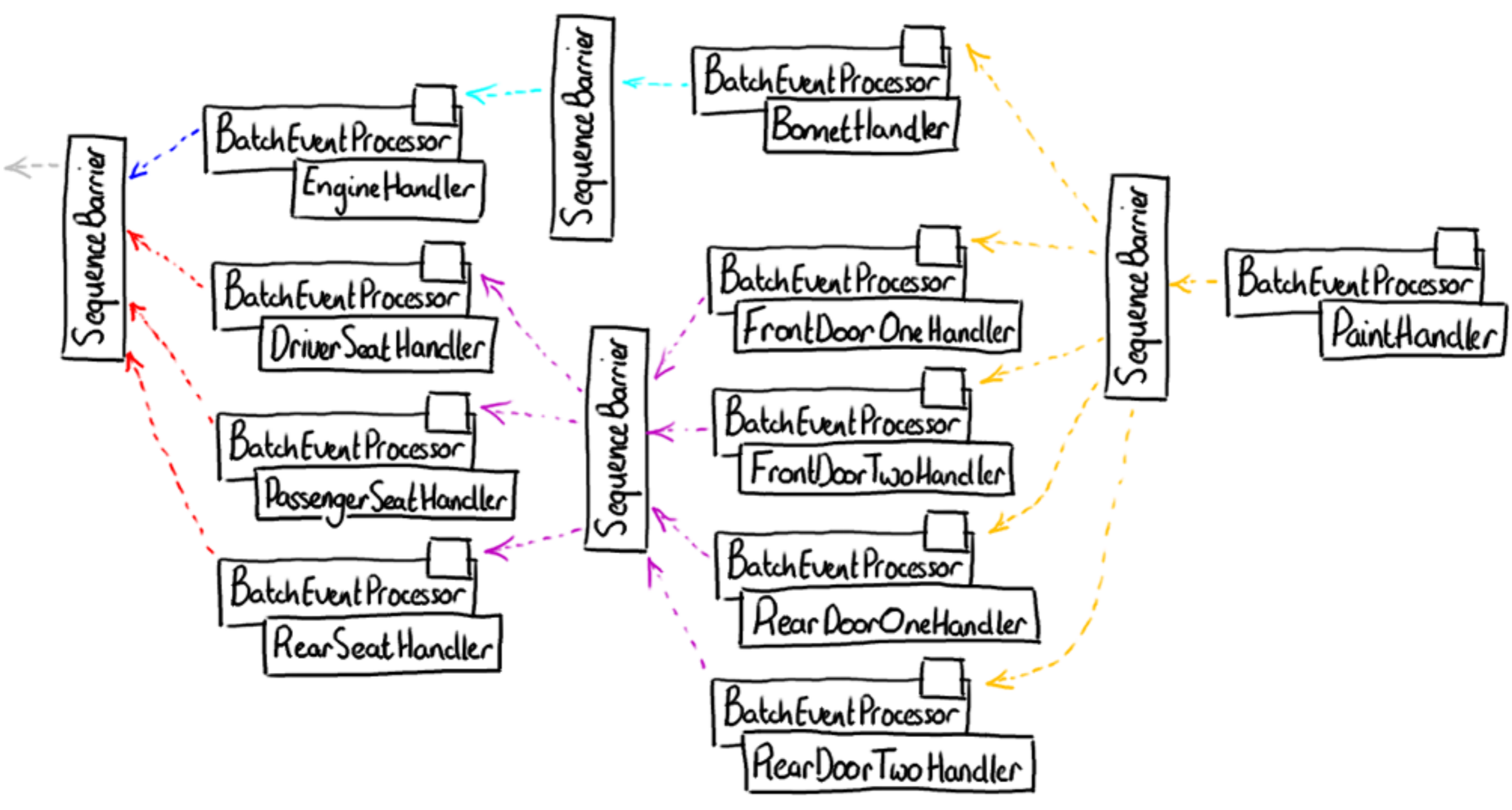
Sequence Barrier

Sequence Barrier

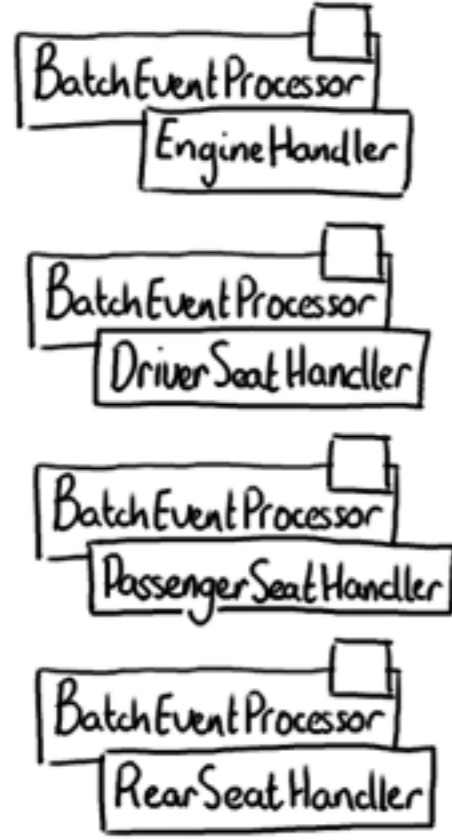


Sequence Barrier



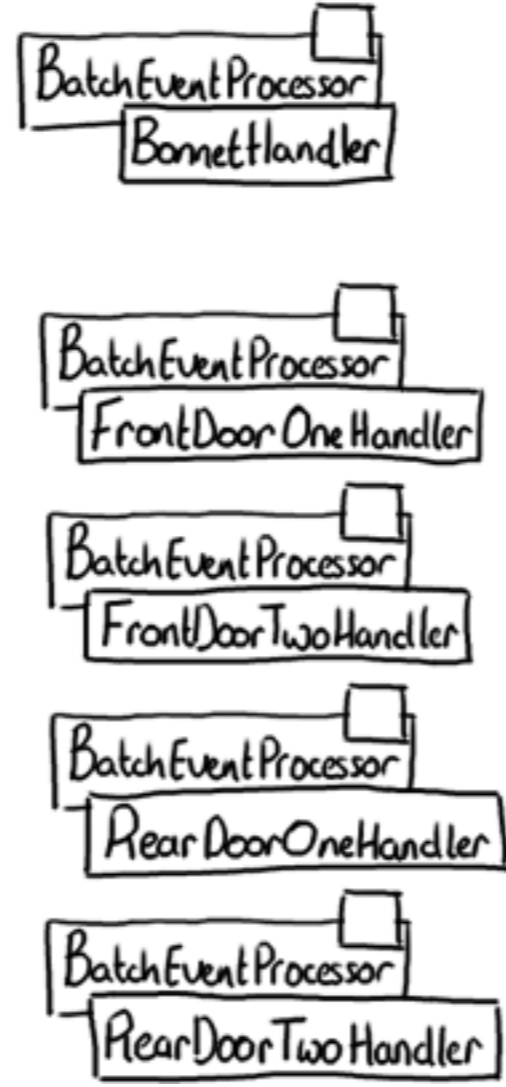


Sequence Barrier



Sequence Barrier

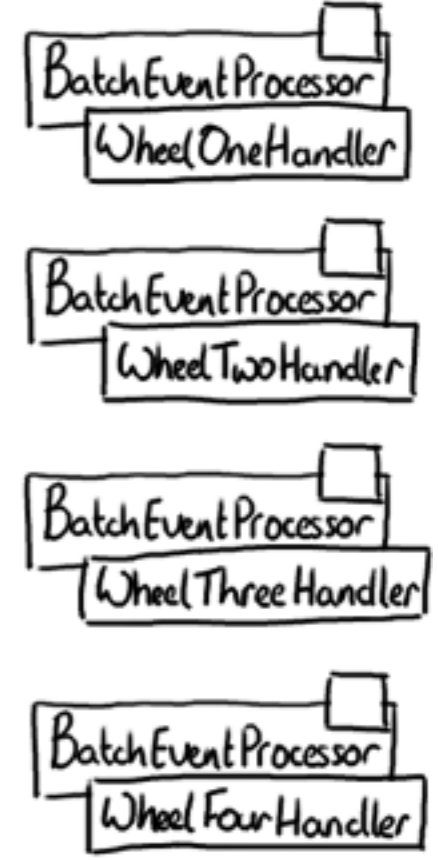
Sequence Barrier



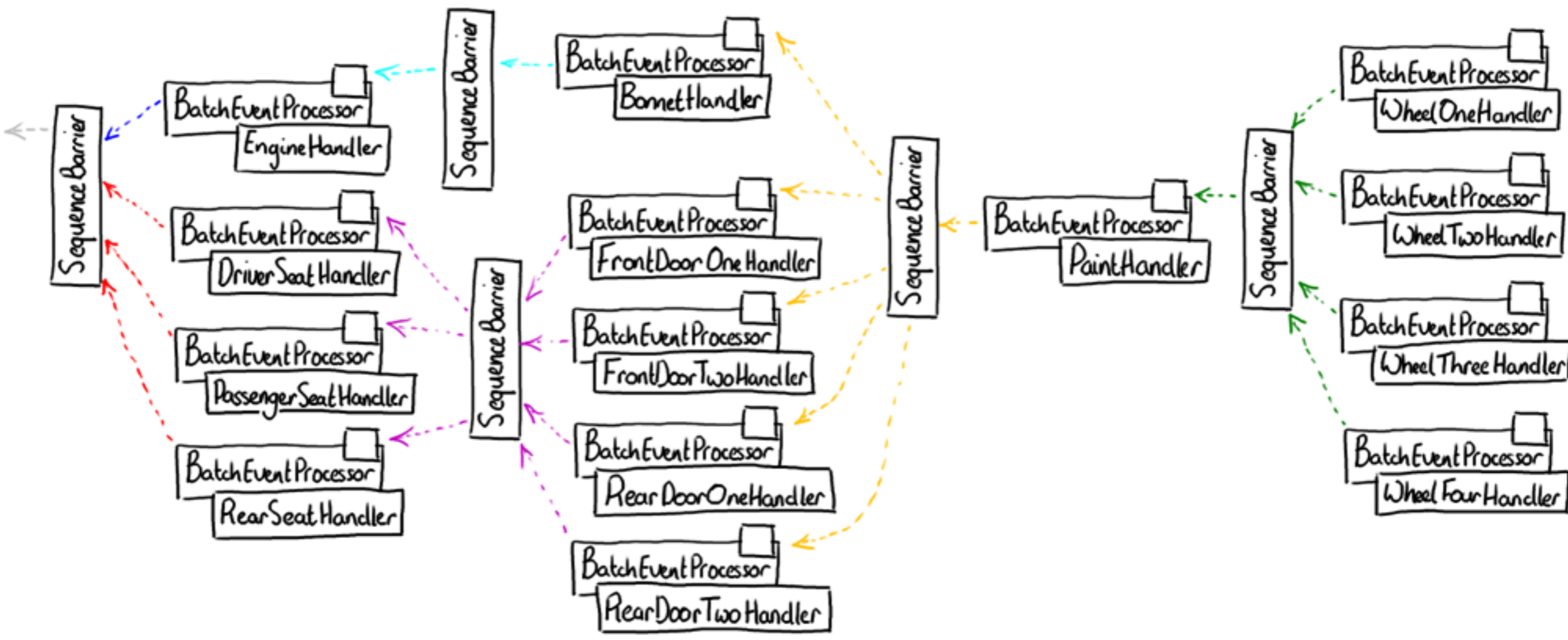
Sequence Barrier

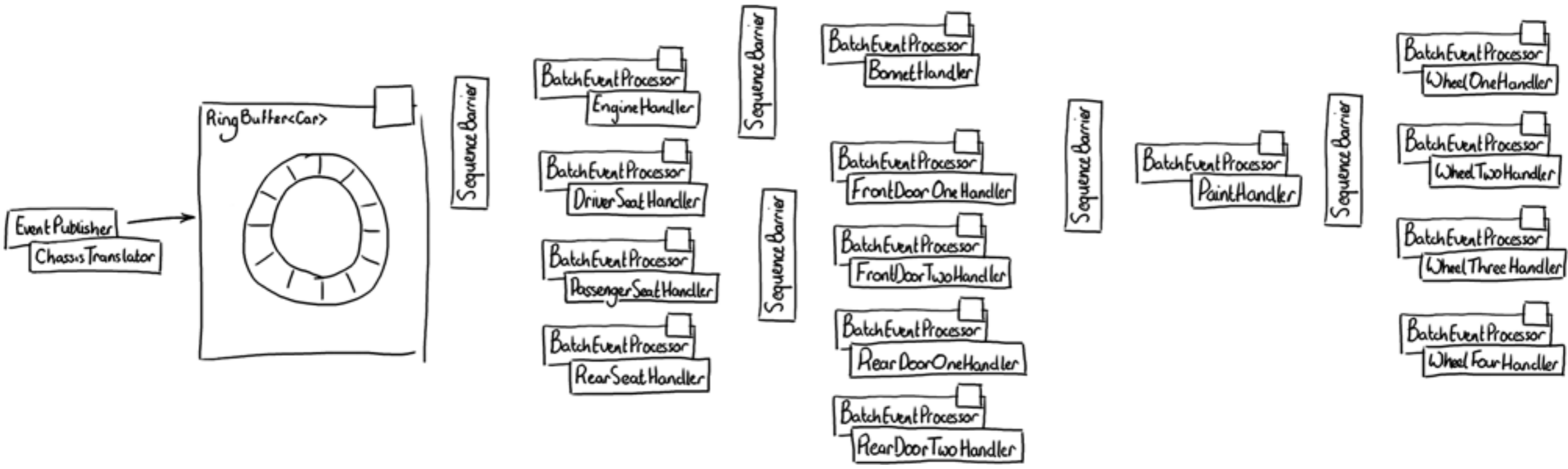


Sequence Barrier

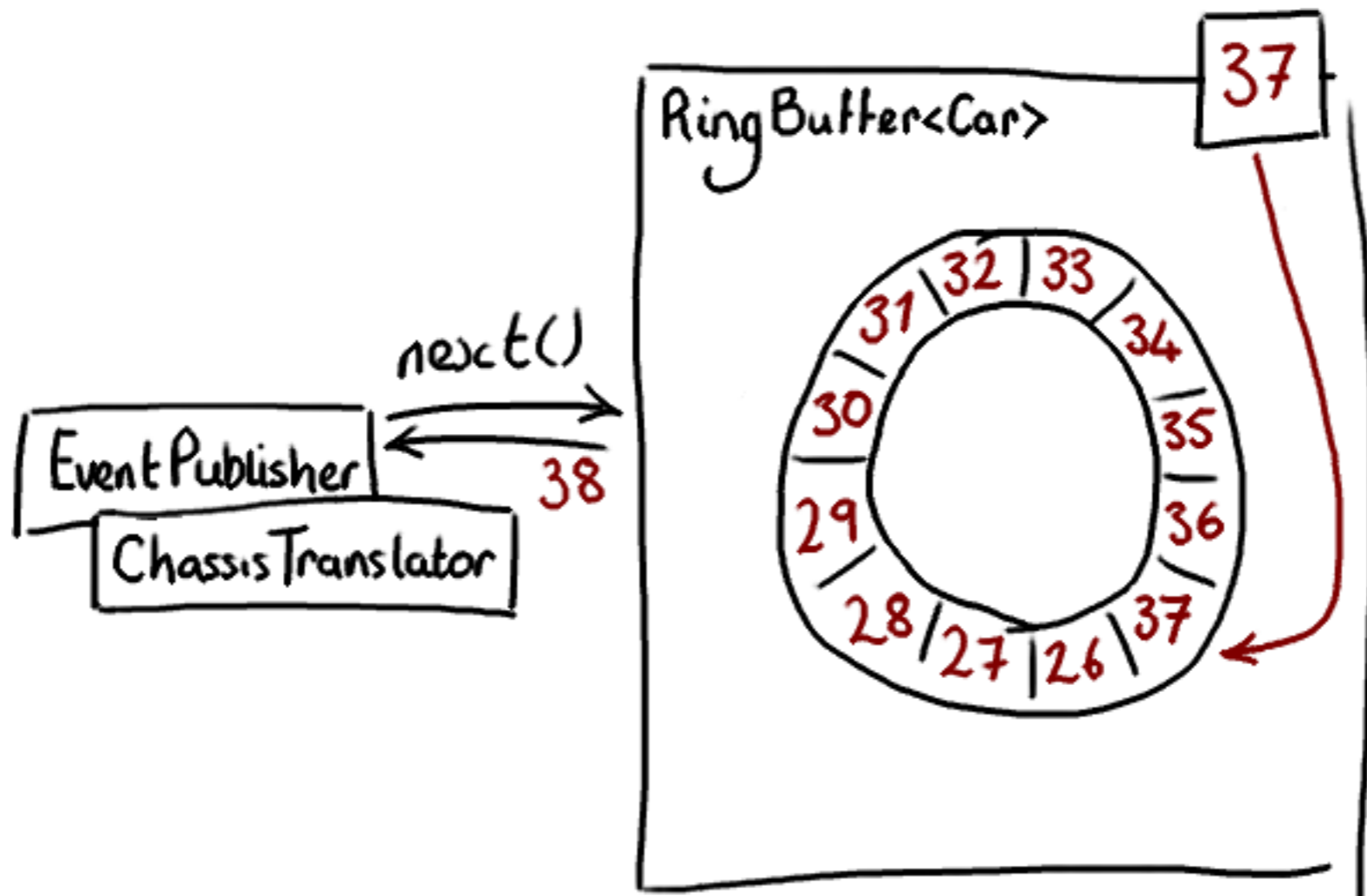


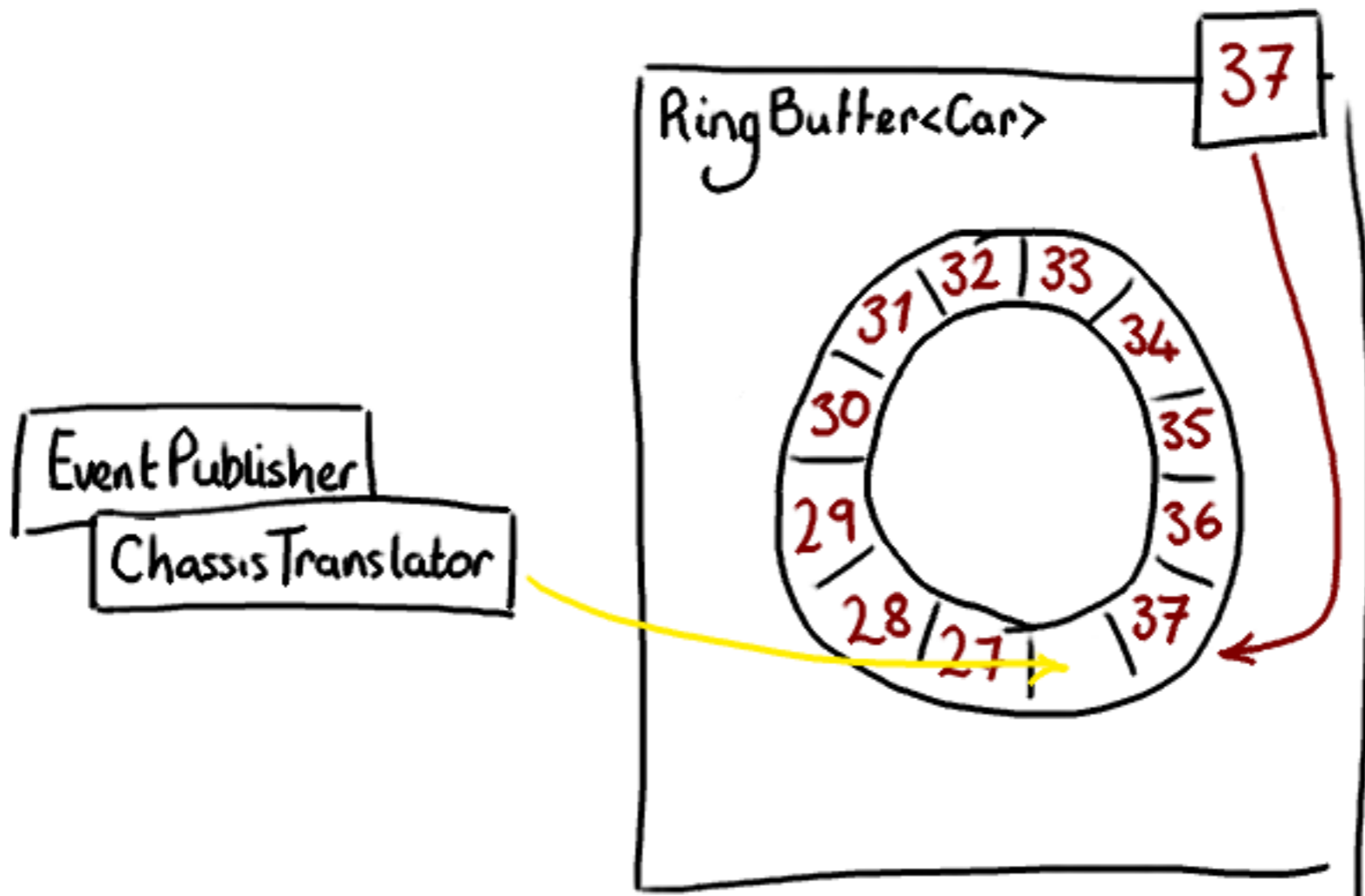


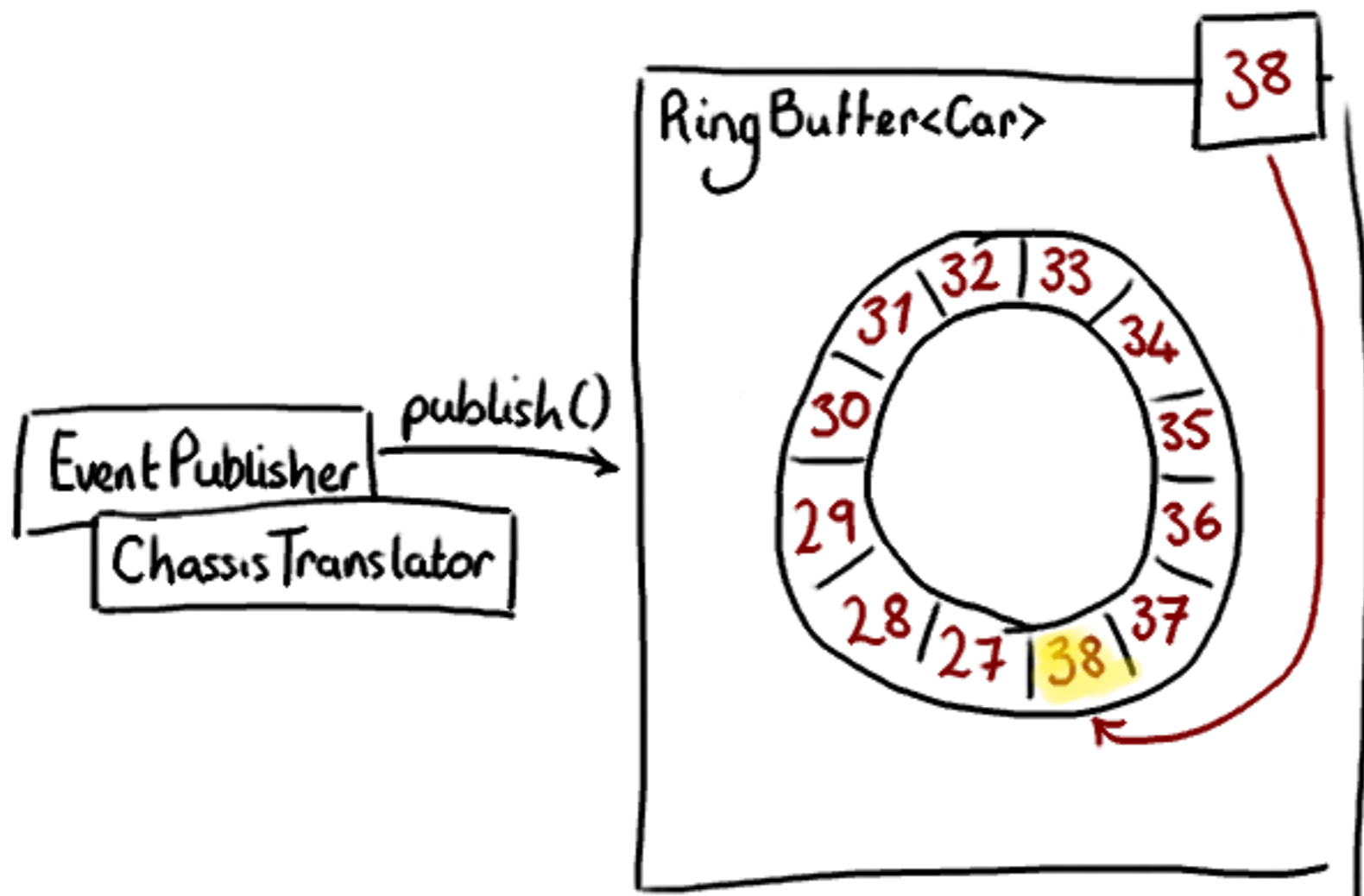


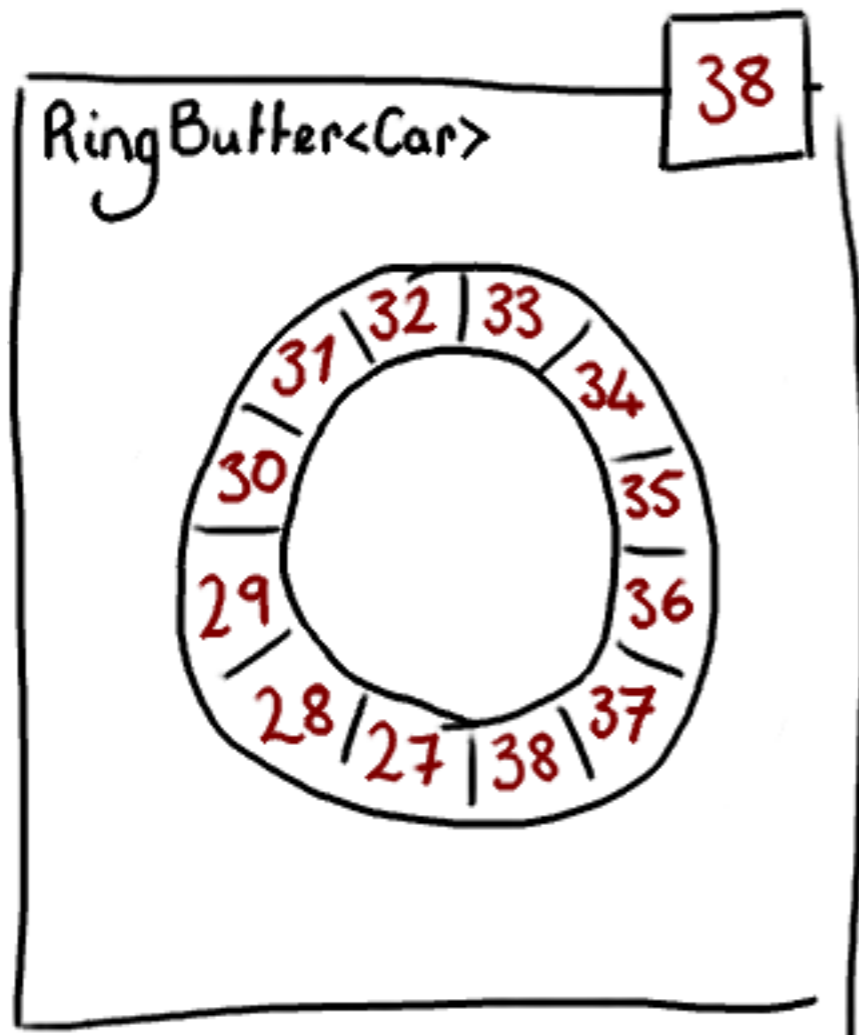




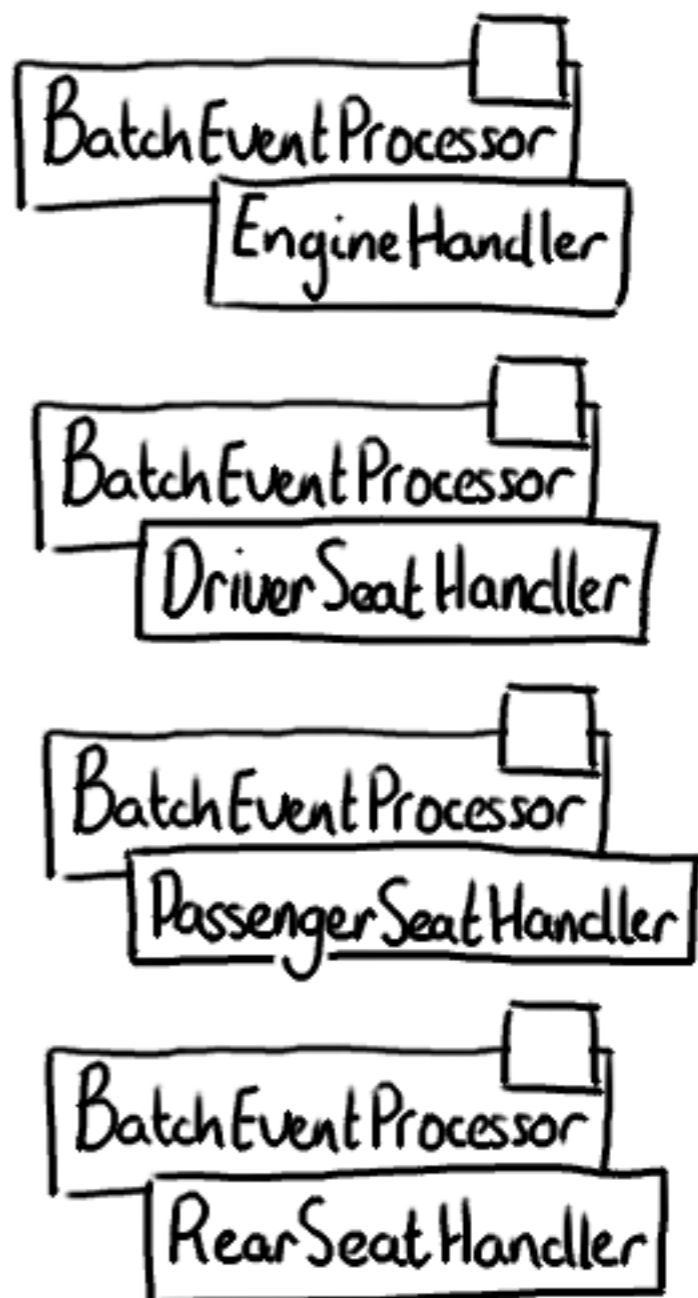


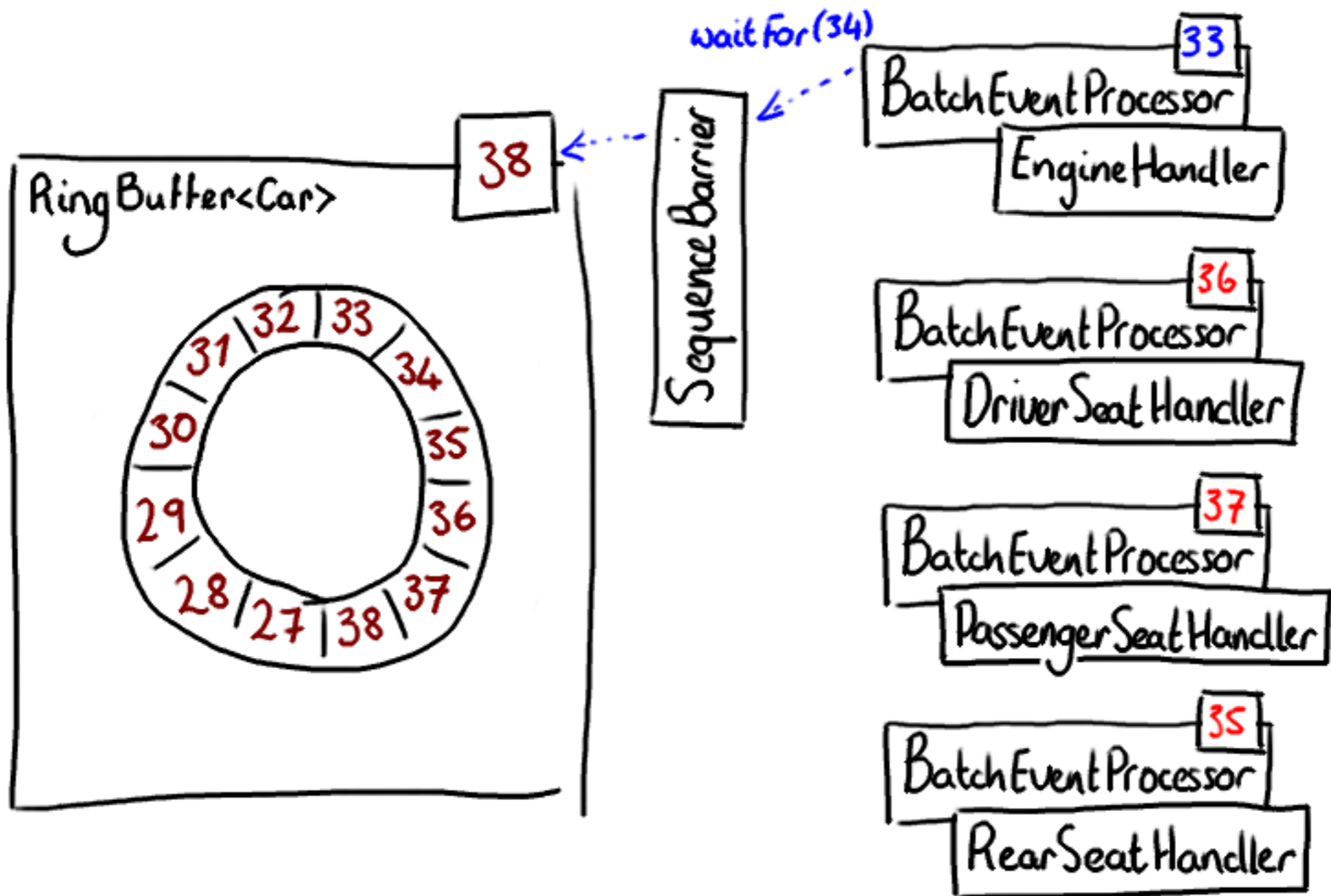


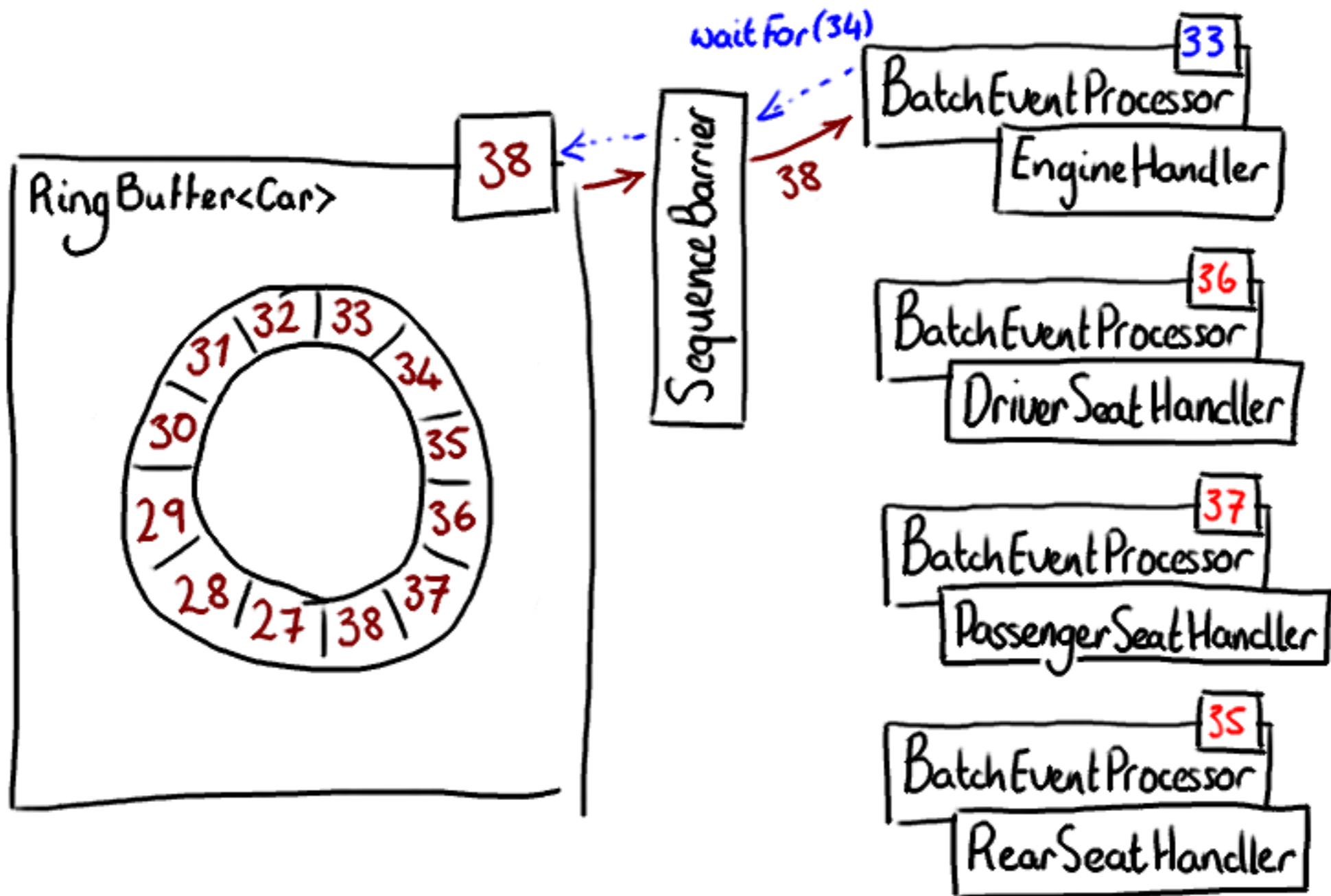


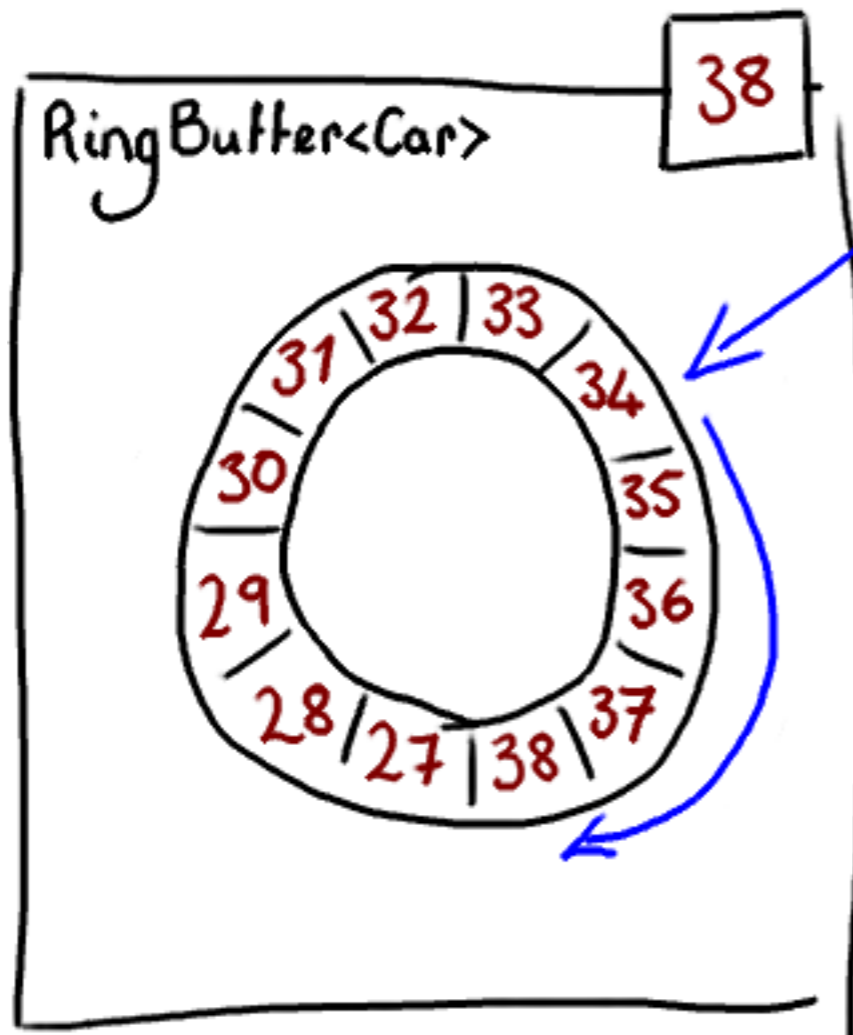


Sequence Barrier

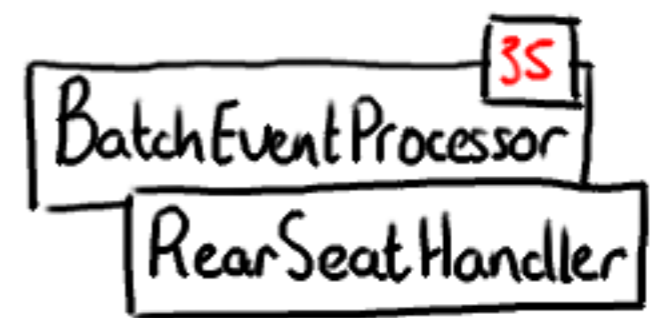
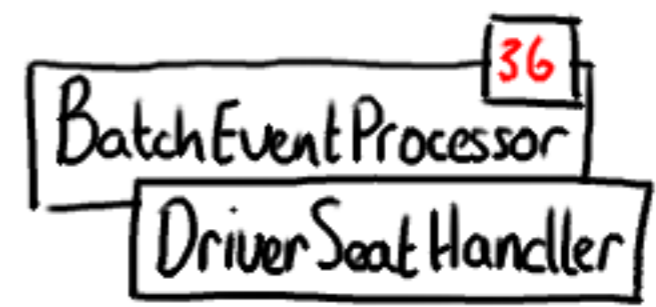
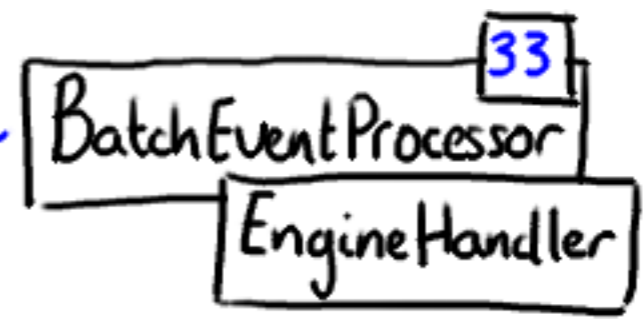




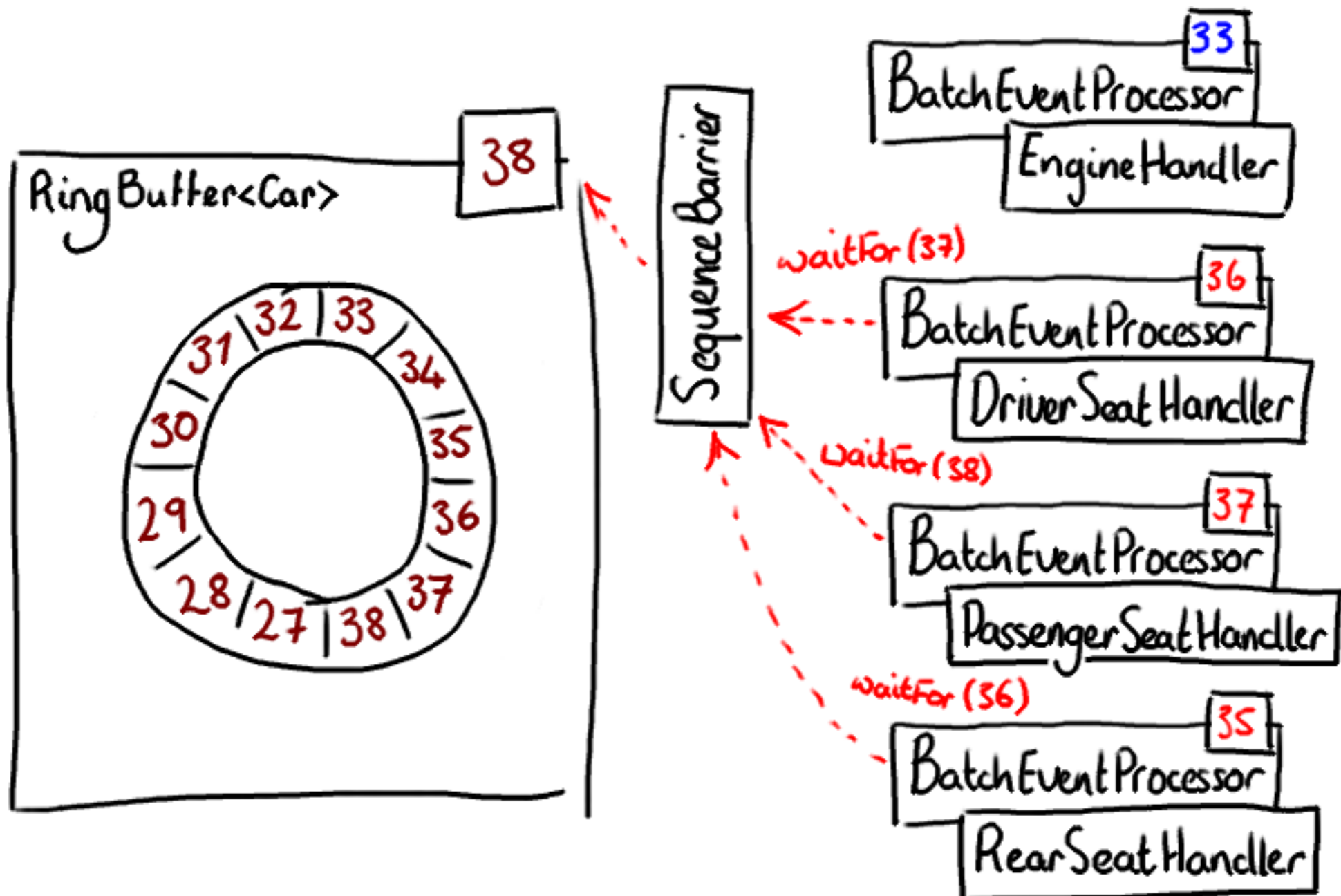




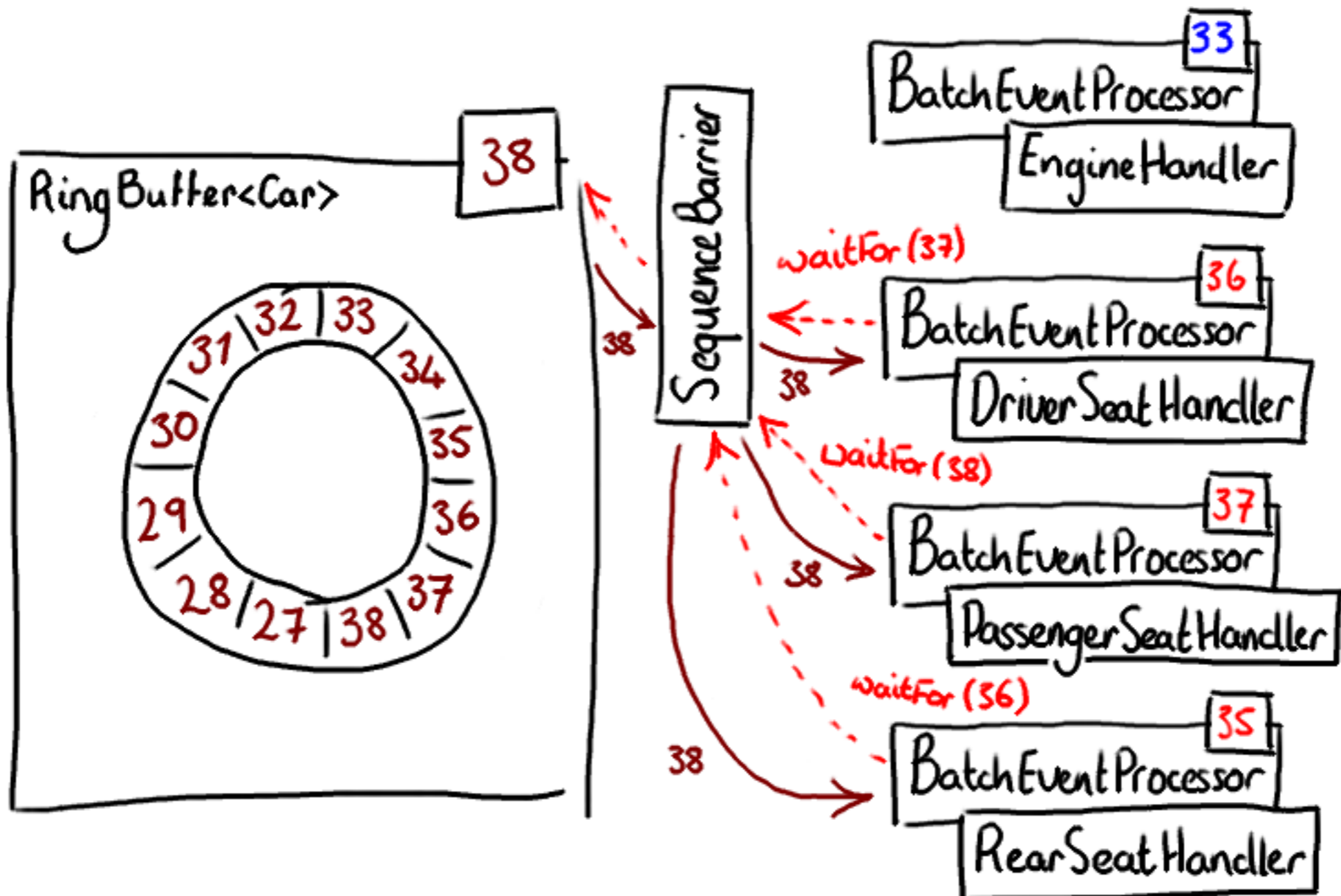
Sequence barrier

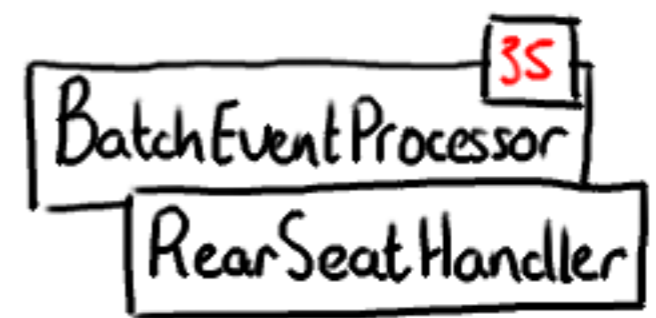
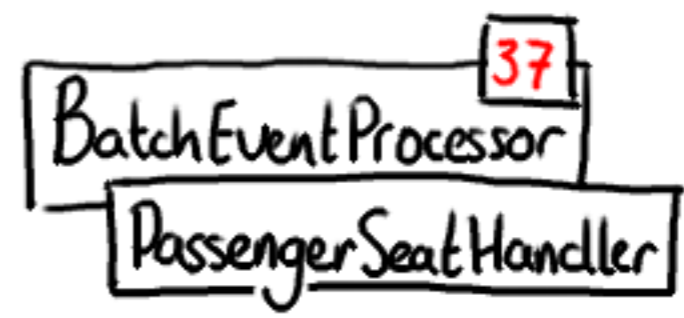
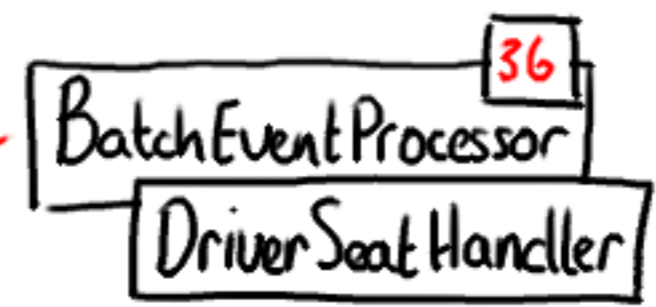
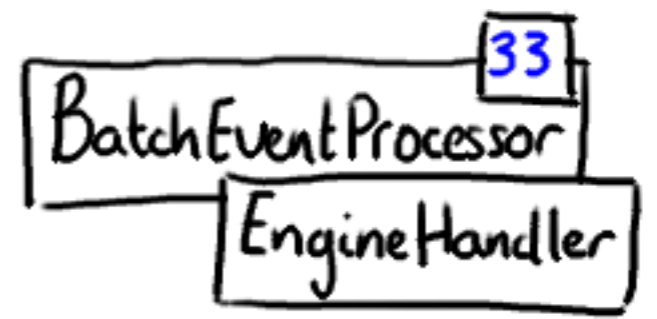
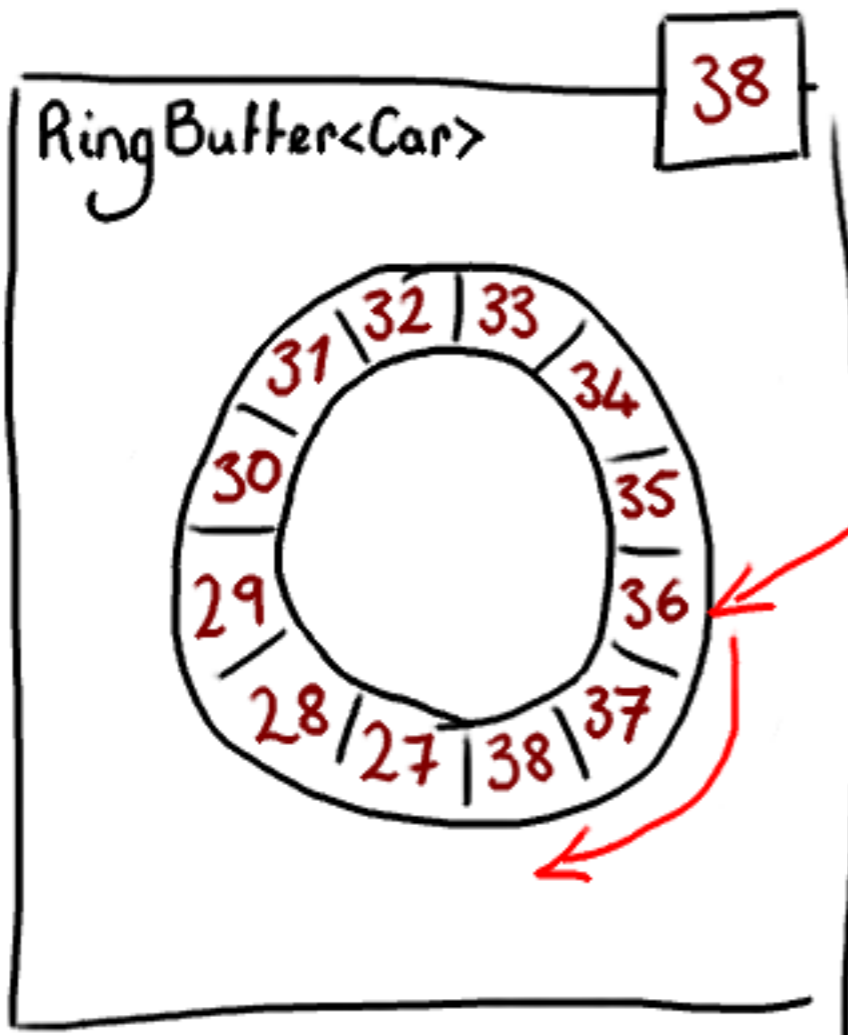


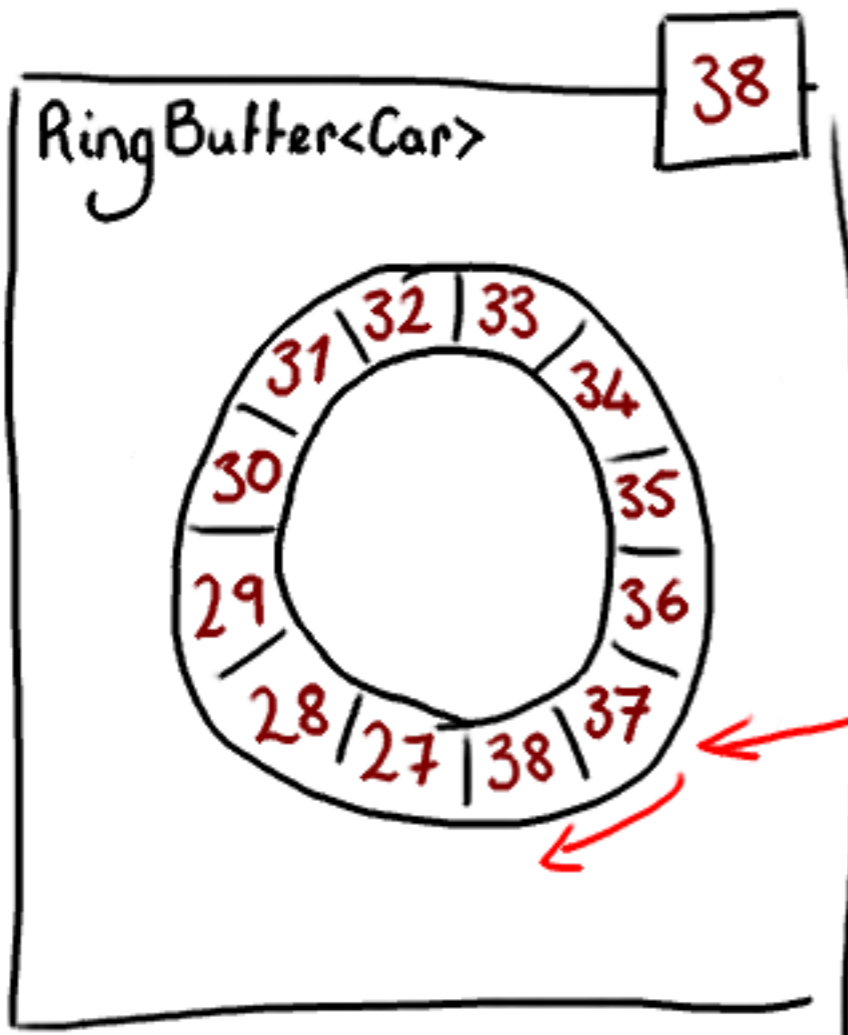




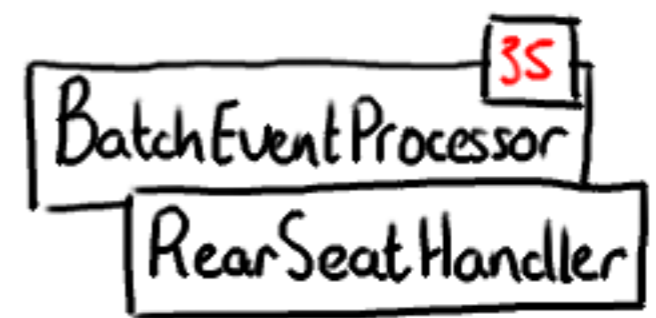
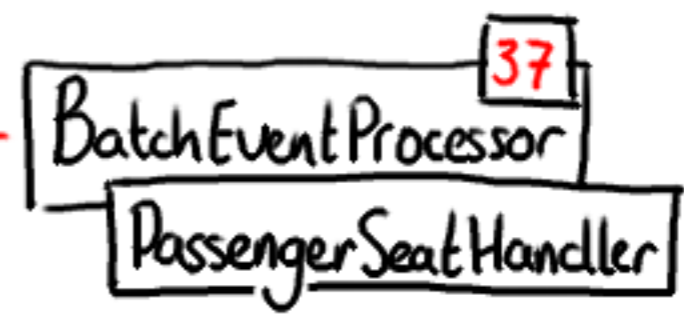
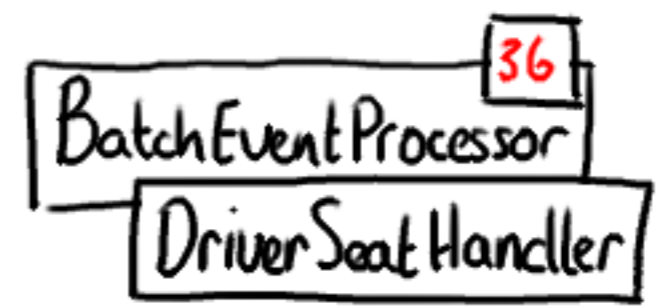
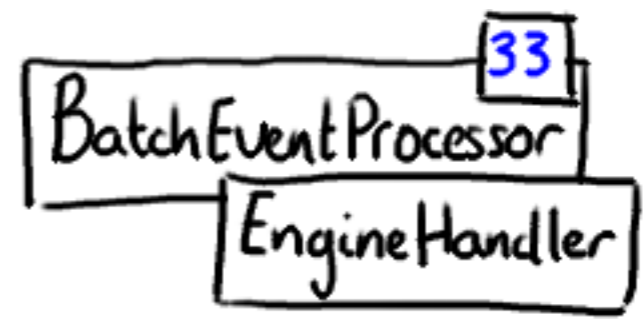


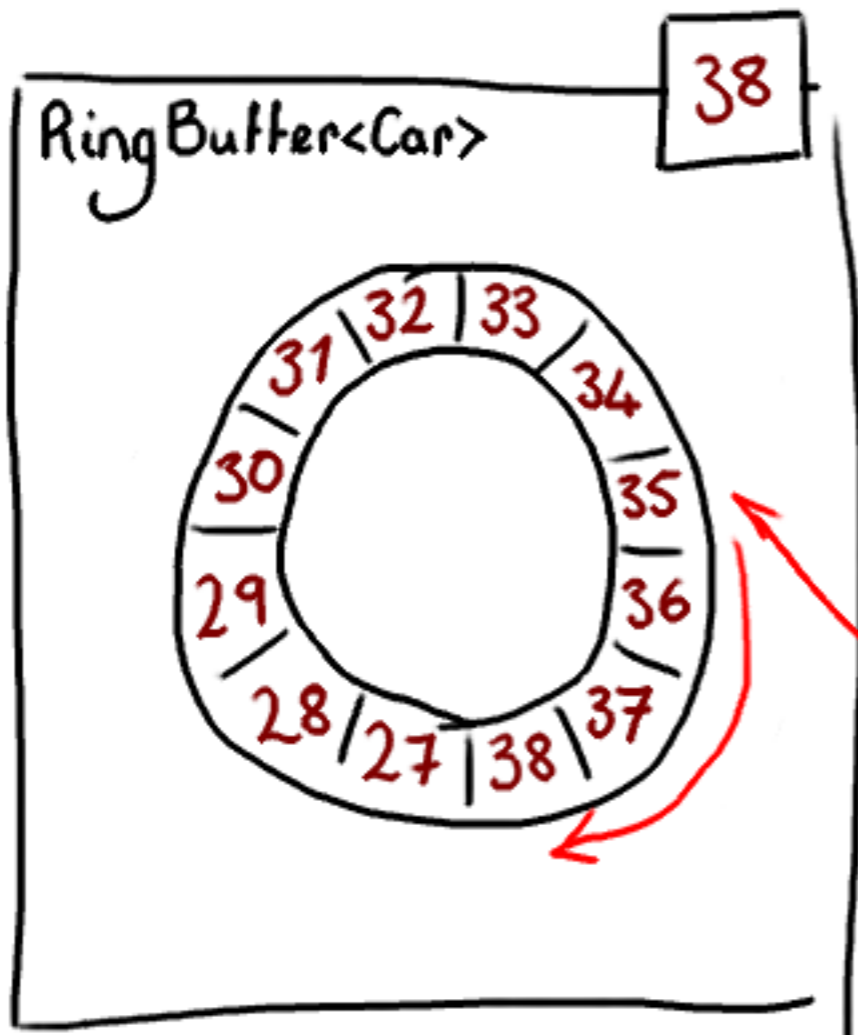




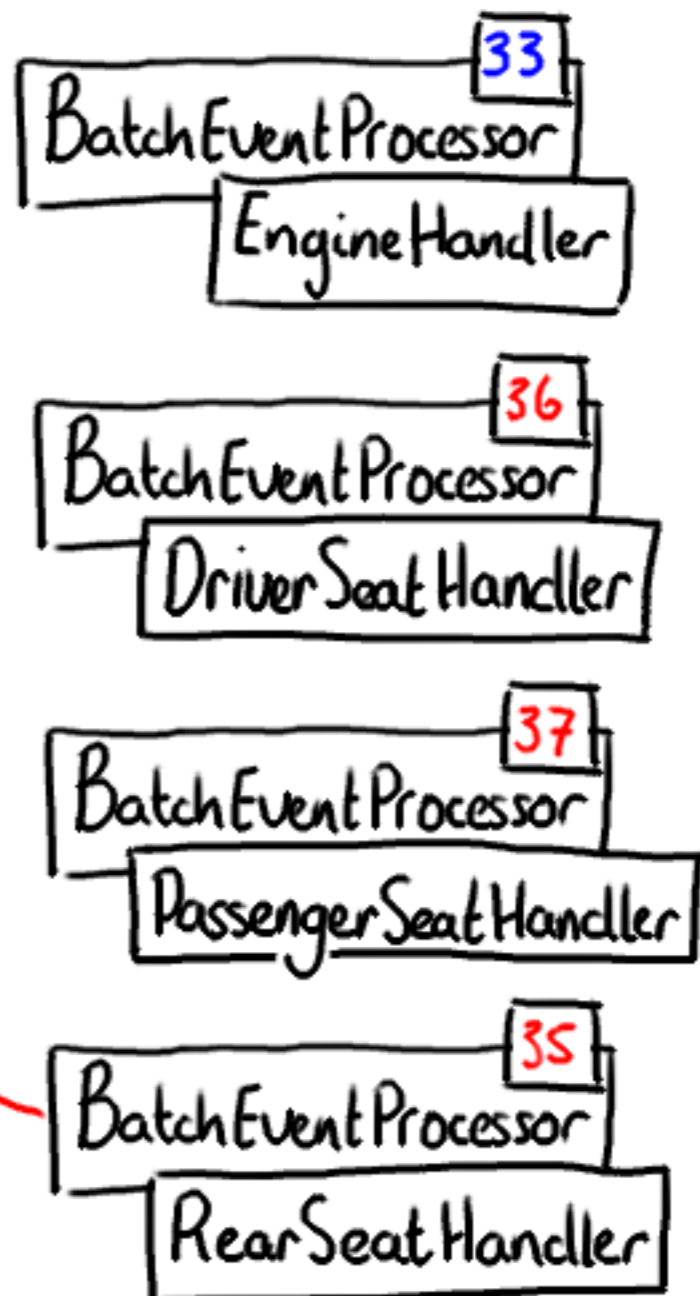


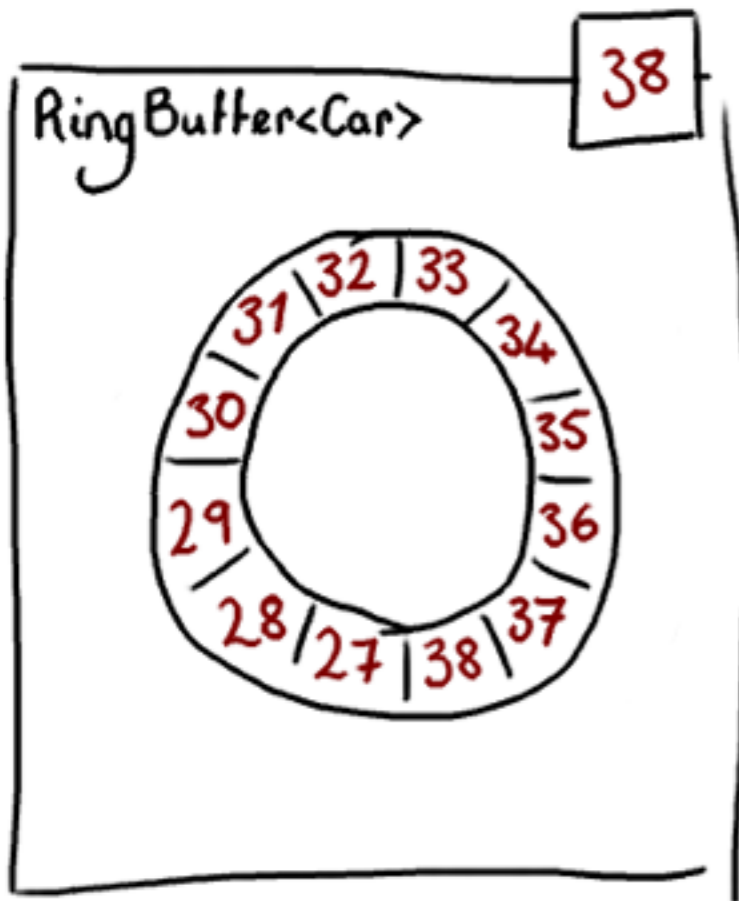
Sequence Barrier



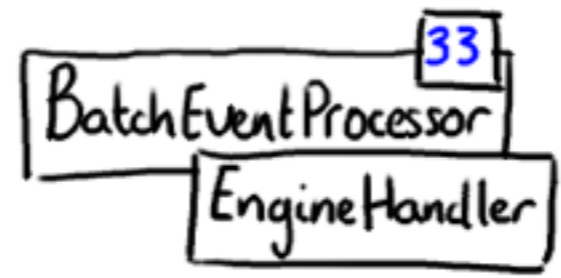


Sequence Barrier

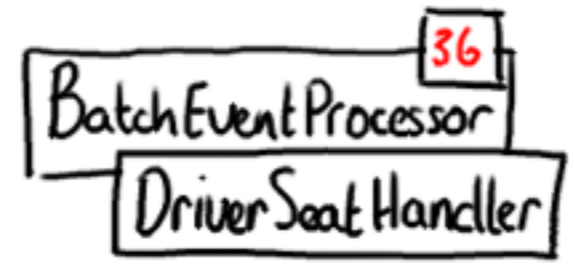
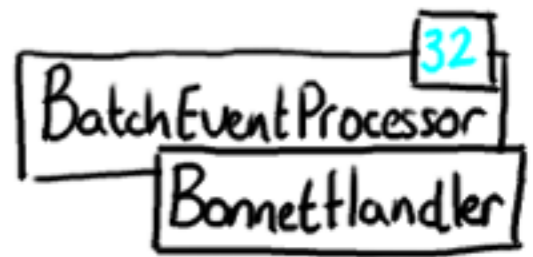




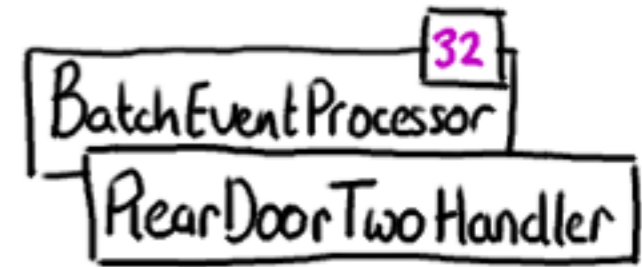
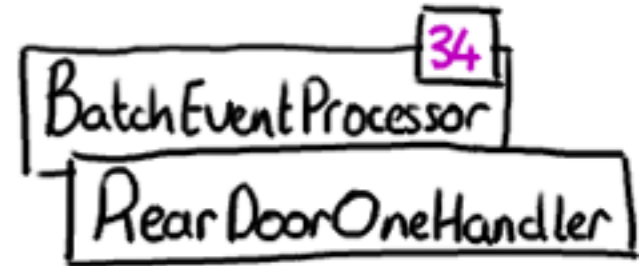
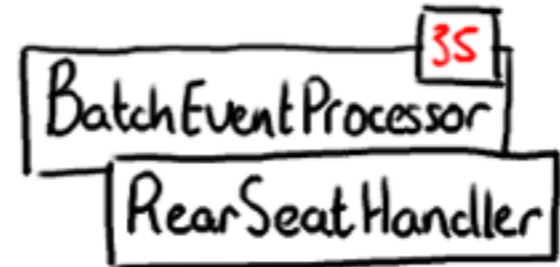
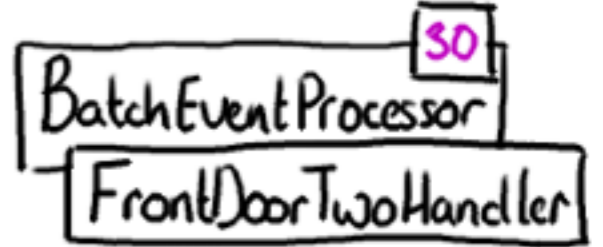
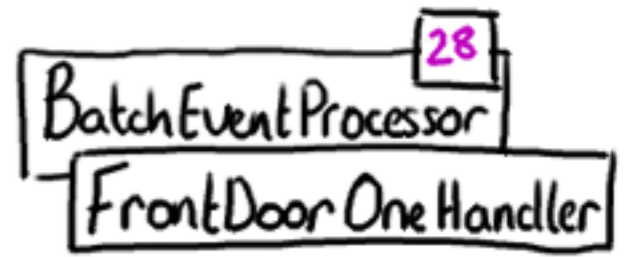
Sequence Barrier



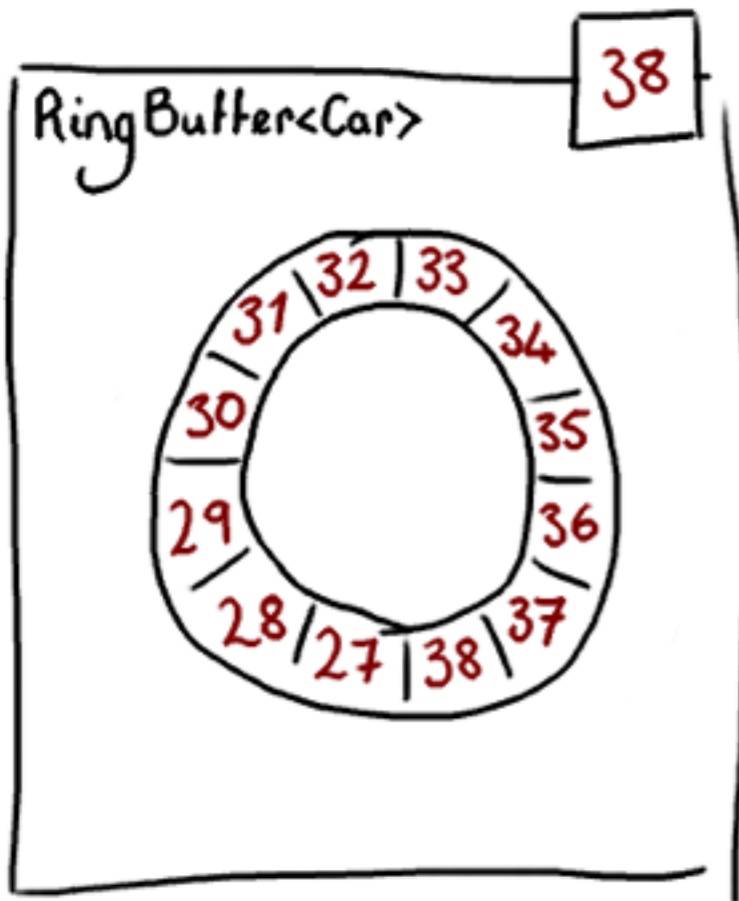
Sequence Barrier



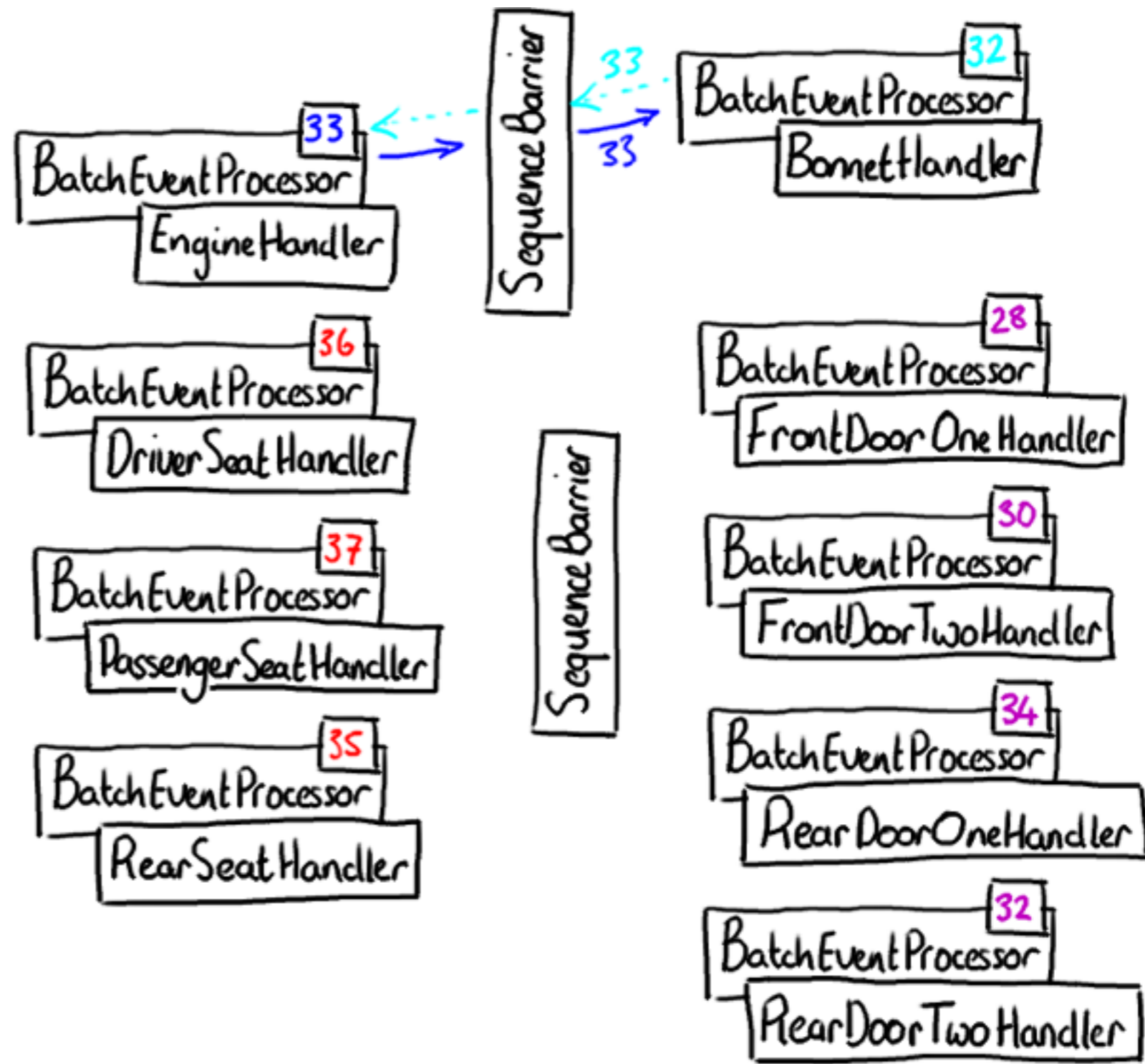
Sequence Barrier

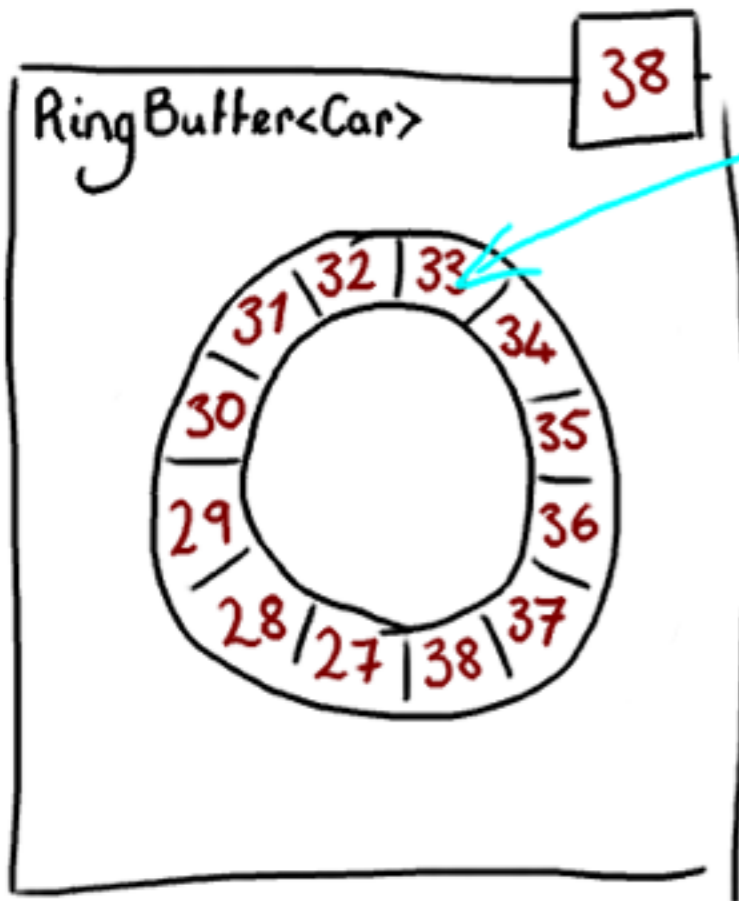




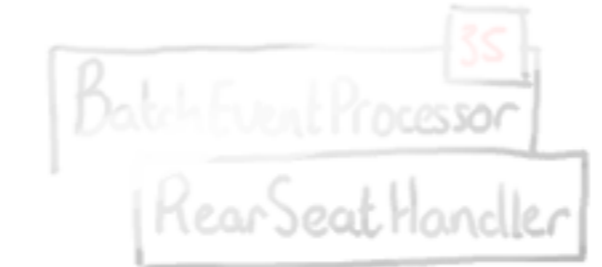
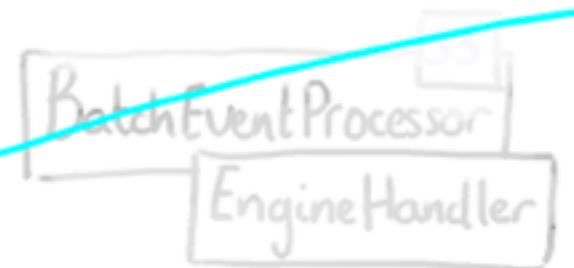


Sequence Barrier



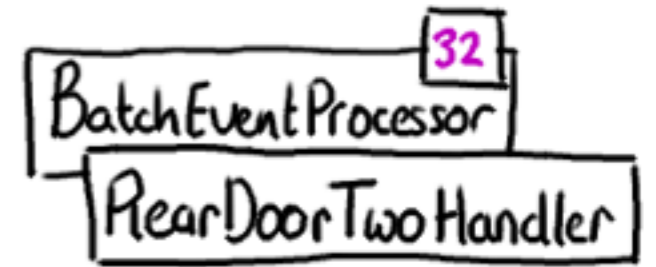
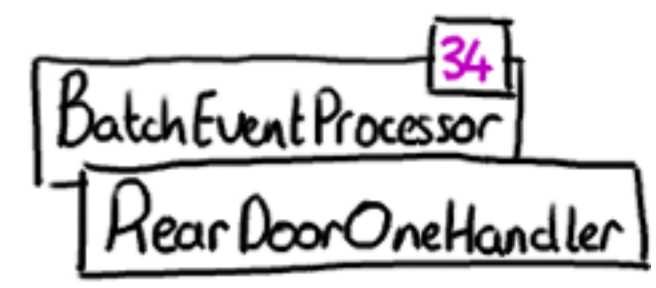
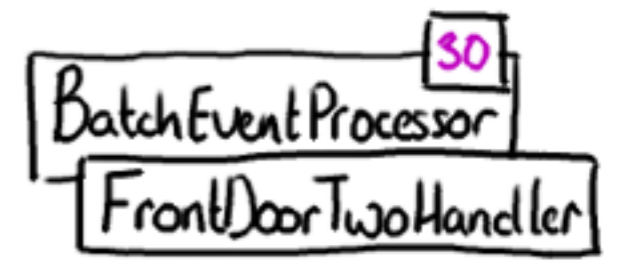
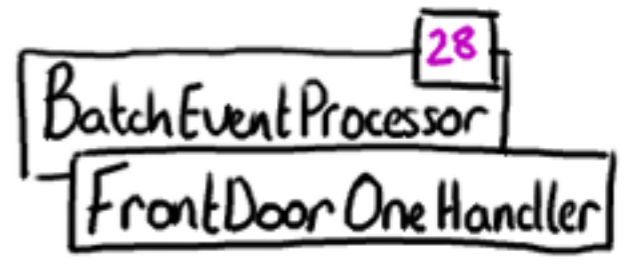


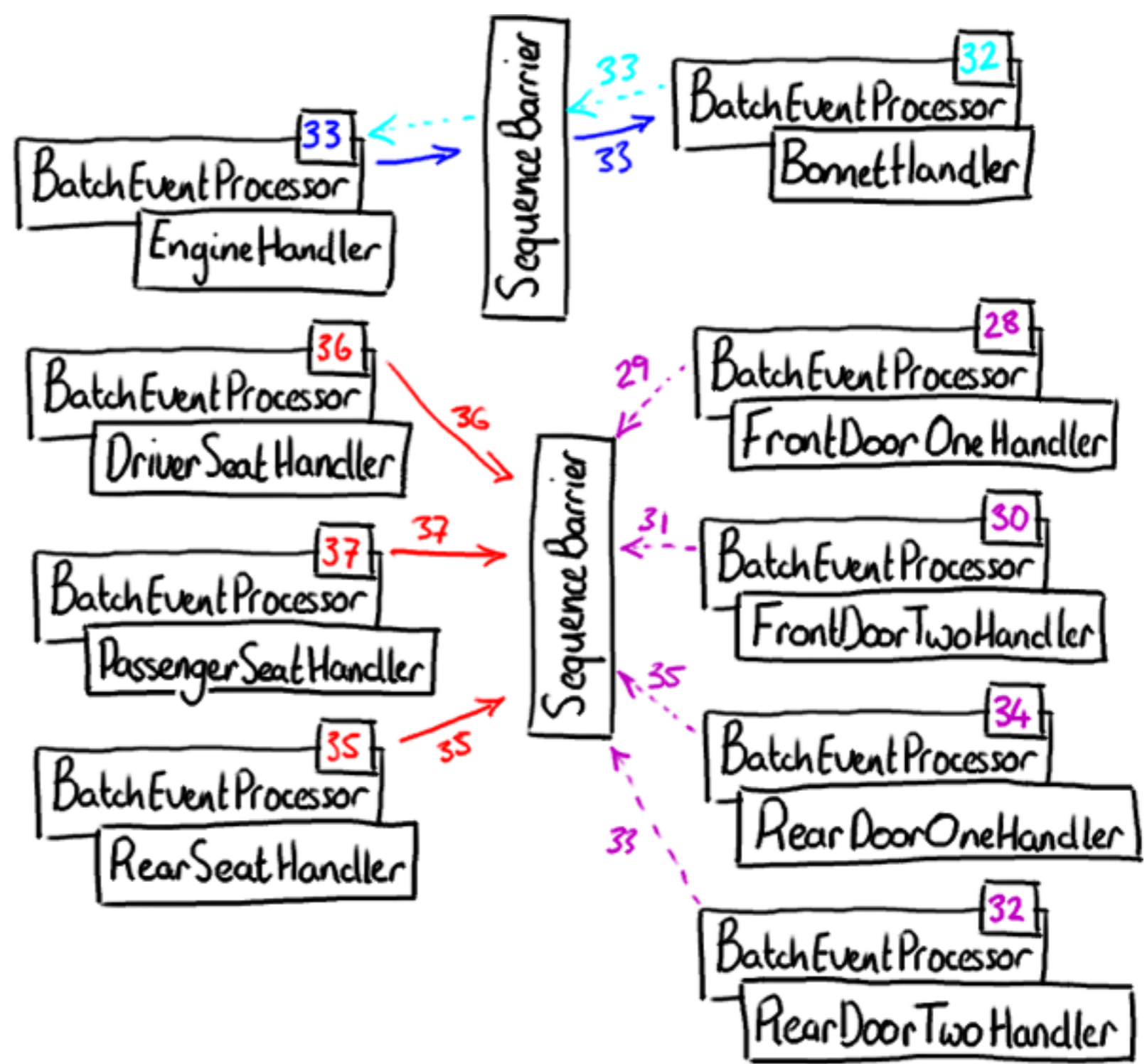
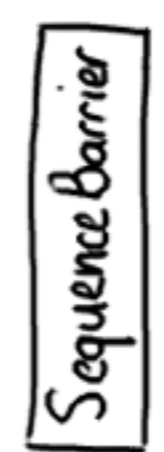
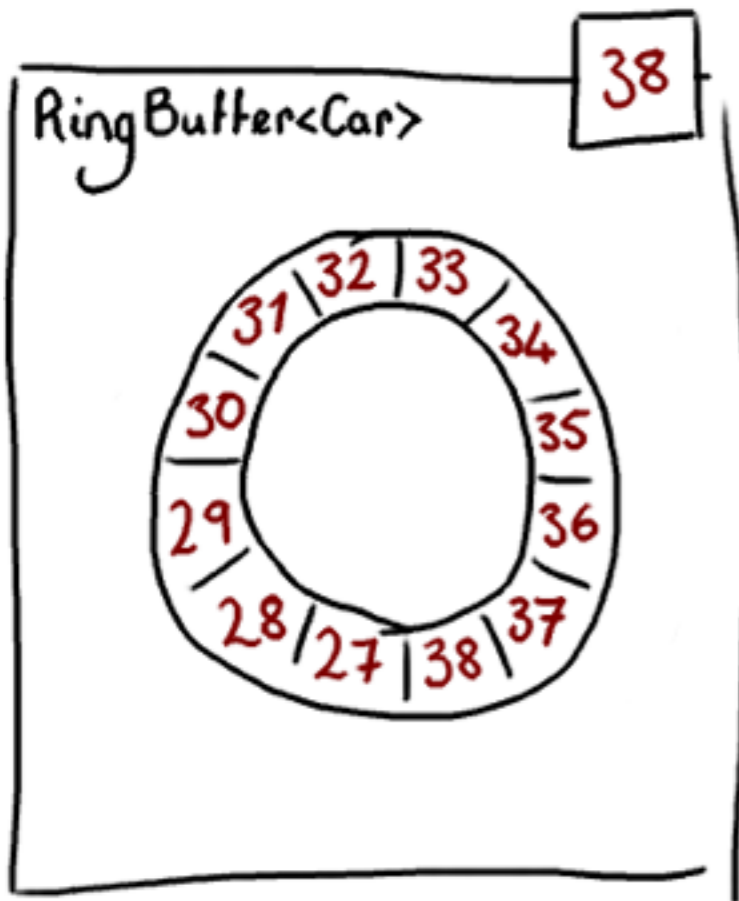
Sequence Barrier



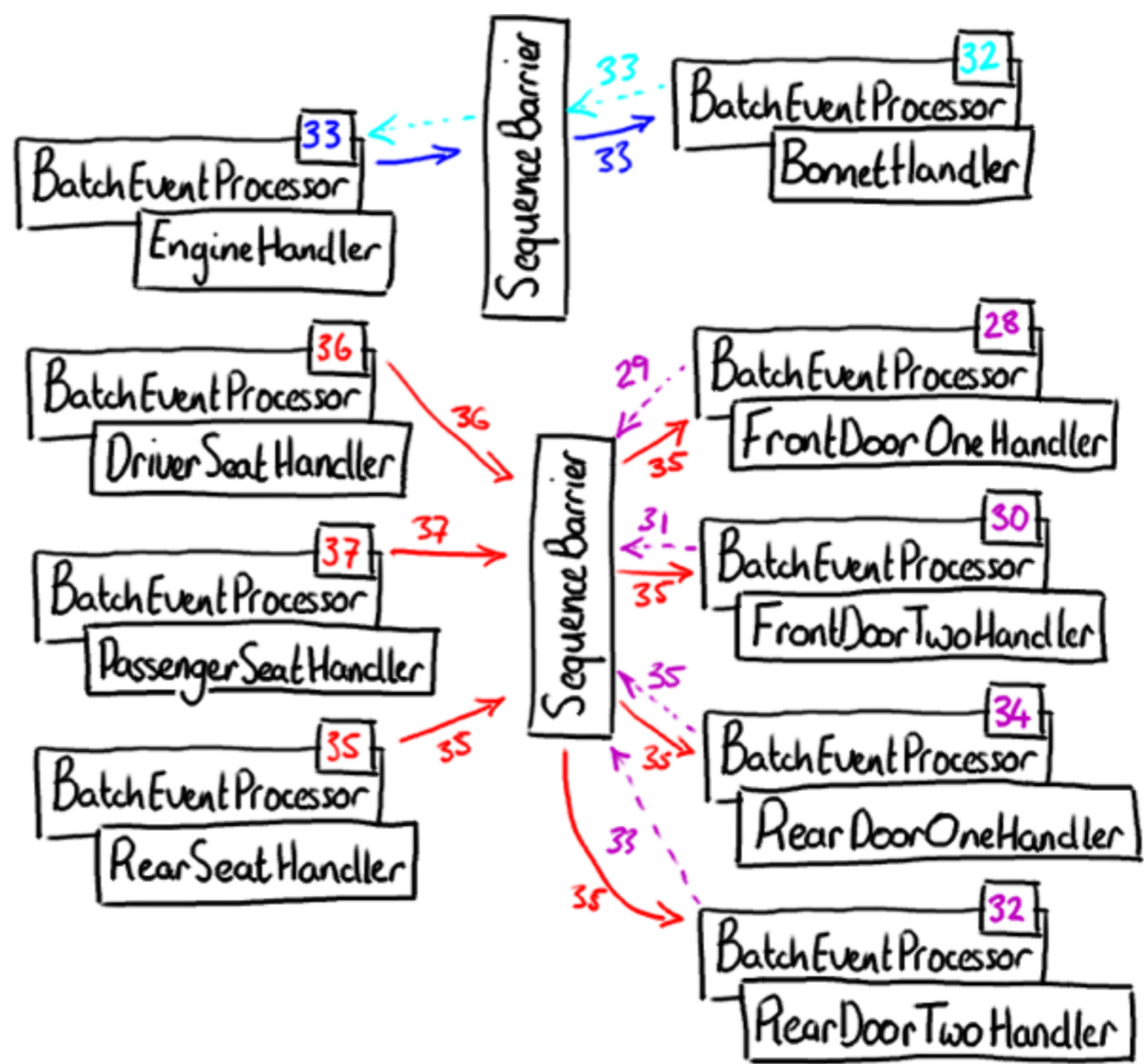
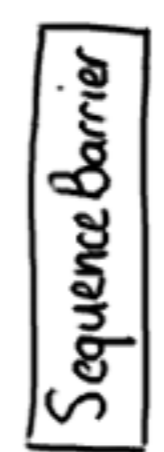
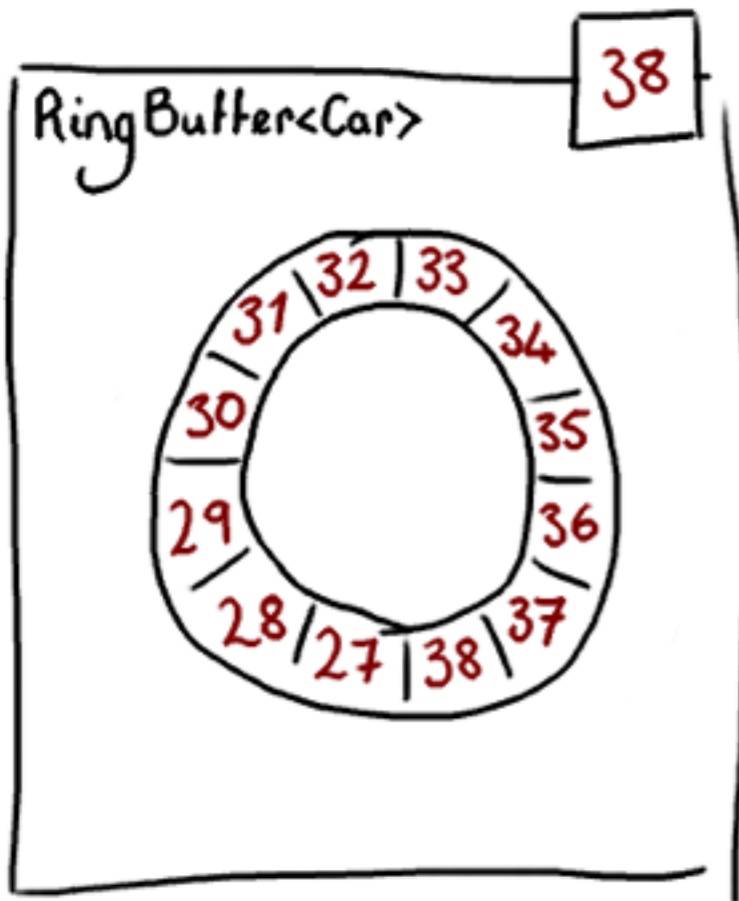
Sequence Barrier

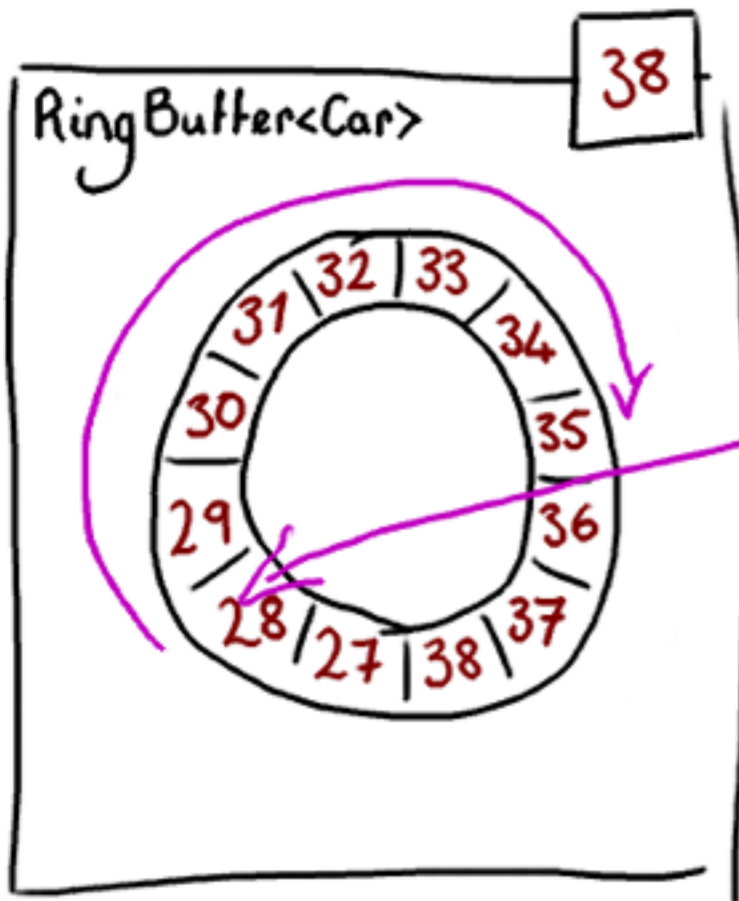
Sequence Barrier











Sequence barrier

BatchEventProcessor 33  
EngineHandler

Sequence barrier

BatchEventProcessor 36  
DriverSeatHandler

Sequence barrier

BatchEventProcessor 37  
PassengerSeatHandler

BatchEventProcessor 35  
RearSeatHandler

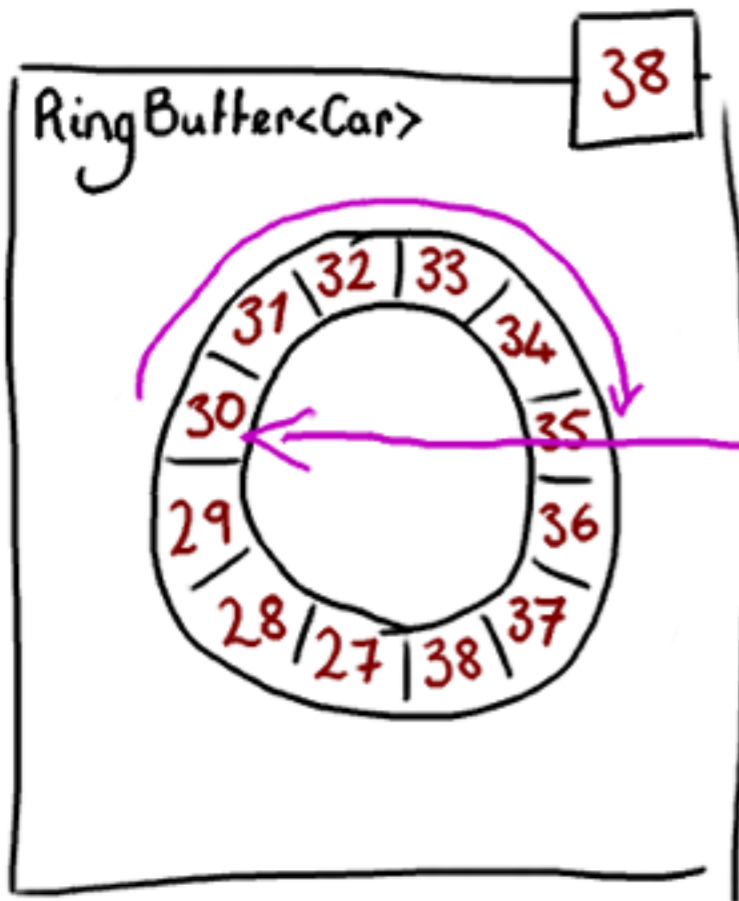
BatchEventProcessor 32  
BonnetHandler

BatchEventProcessor 28  
FrontDoorOneHandler

BatchEventProcessor 30  
FrontDoorTwoHandler

BatchEventProcessor 34  
RearDoorOneHandler

BatchEventProcessor 32  
RearDoorTwoHandler



Sequence barrier

BatchEventProcessor 33  
EngineHandler

Sequence barrier

BatchEventProcessor 32  
BonnetHandler

BatchEventProcessor 36  
DriverSeatHandler

Sequence barrier

BatchEventProcessor 28  
FrontDoorOneHandler

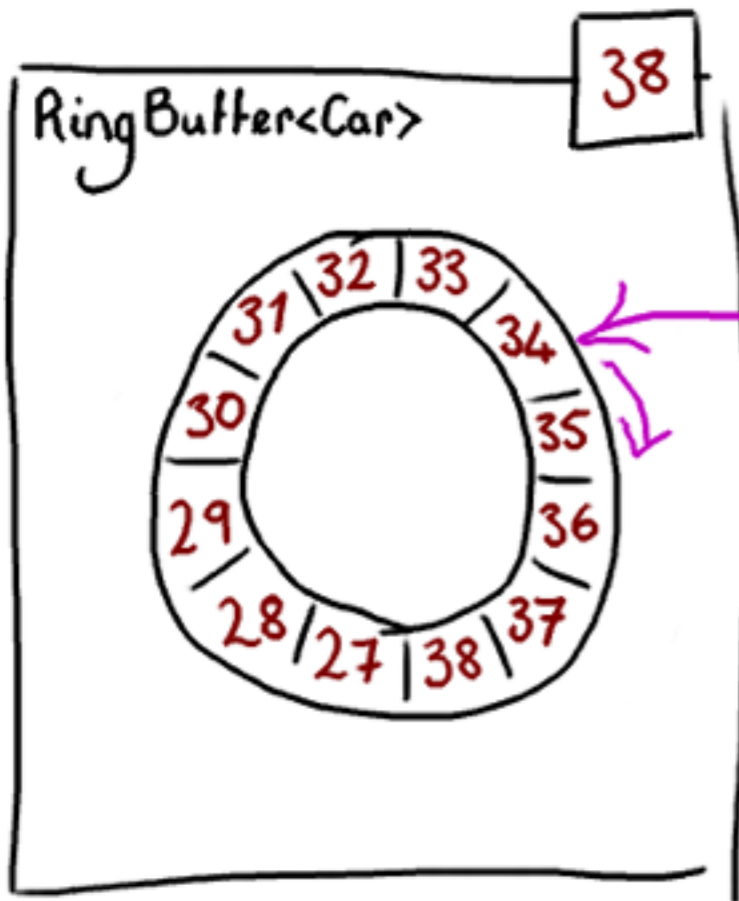
BatchEventProcessor 37  
PassengerSeatHandler

BatchEventProcessor 30  
FrontDoorTwoHandler

BatchEventProcessor 35  
RearSeatHandler

BatchEventProcessor 34  
RearDoorOneHandler

BatchEventProcessor 32  
RearDoorTwoHandler

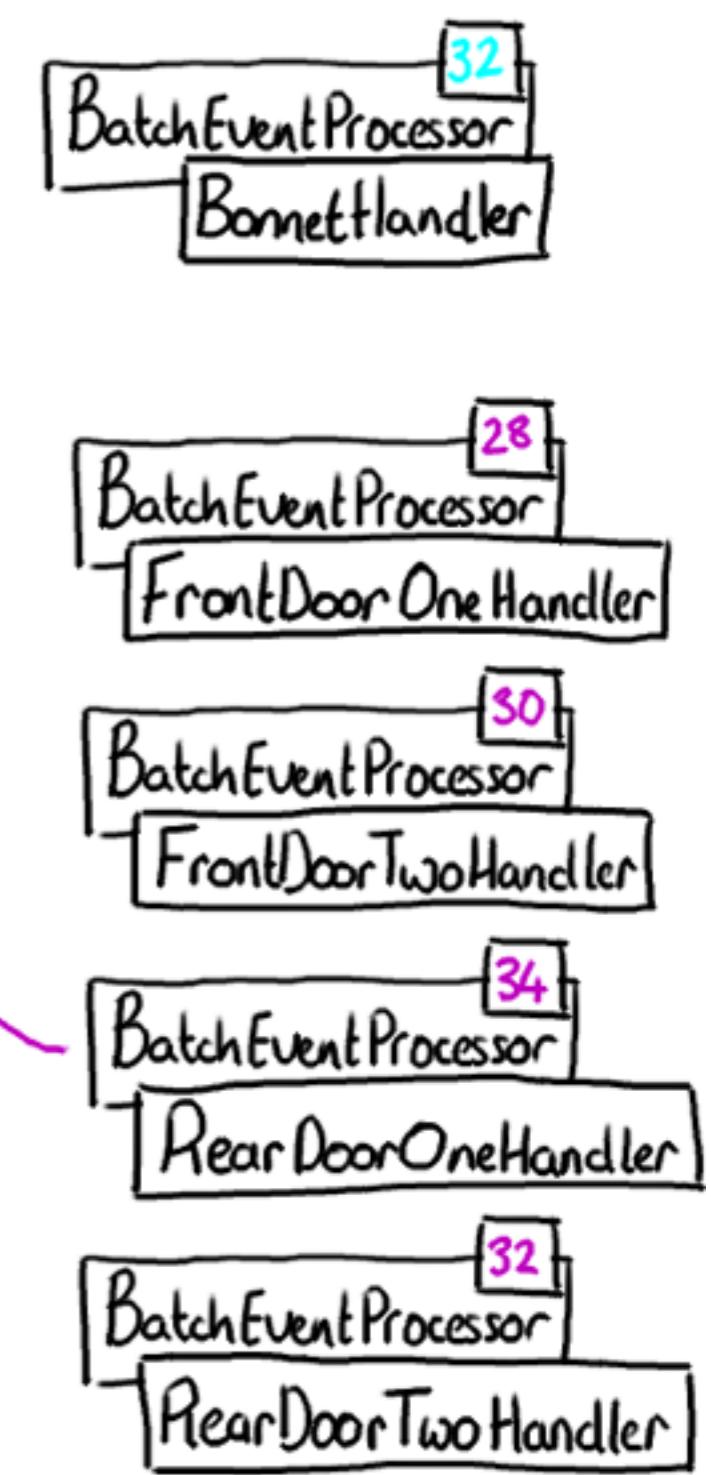


Sequence Barrier

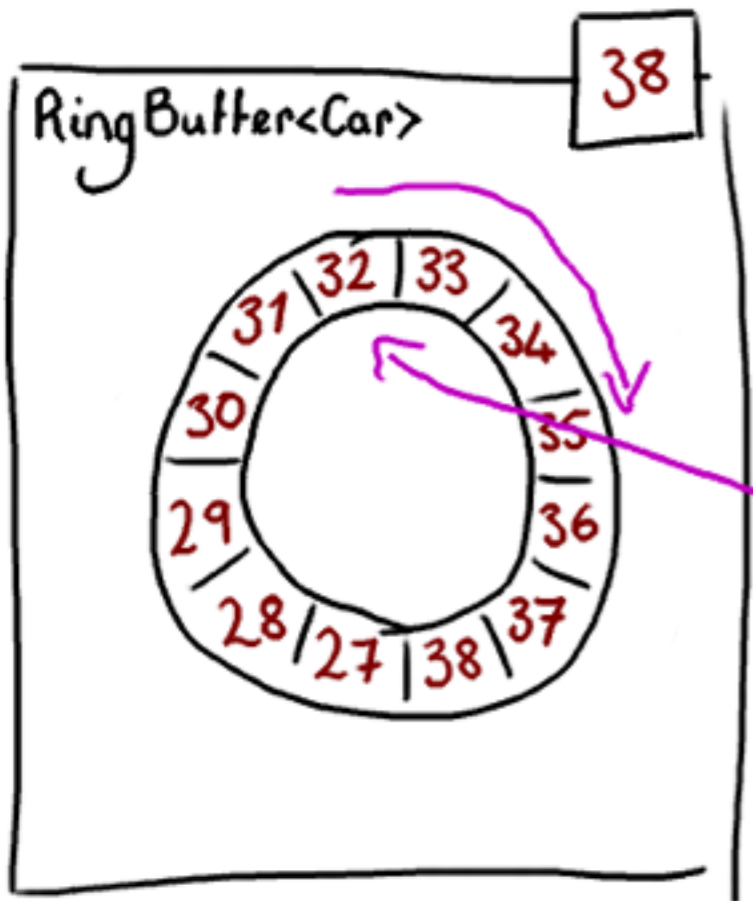


Sequence Barrier

Sequence Barrier







Sequence barrier

33  
BatchEventProcessor  
EngineHandler

Sequence barrier

32  
BatchEventProcessor  
BonnetHandler

36  
BatchEventProcessor  
DriverSeatHandler

Sequence barrier

28  
BatchEventProcessor  
FrontDoorOneHandler

37  
BatchEventProcessor  
PassengerSeatHandler

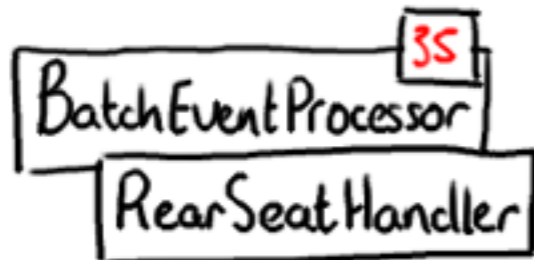
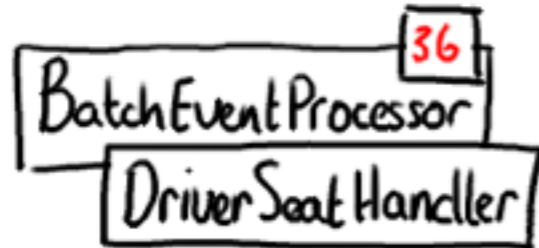
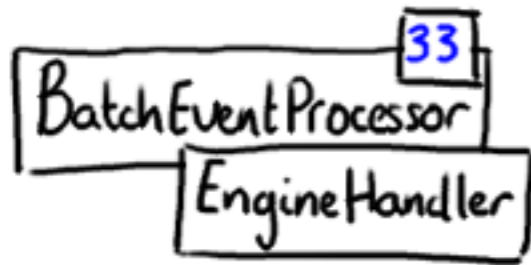
30  
BatchEventProcessor  
FrontDoorTwoHandler

35  
BatchEventProcessor  
RearSeatHandler

34  
BatchEventProcessor  
RearDoorOneHandler

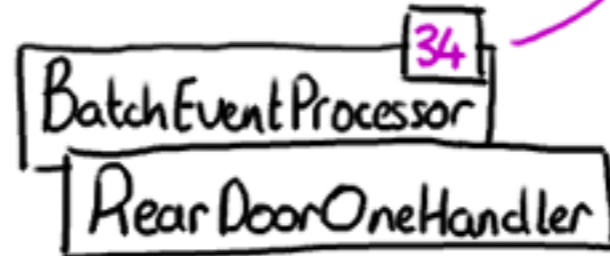
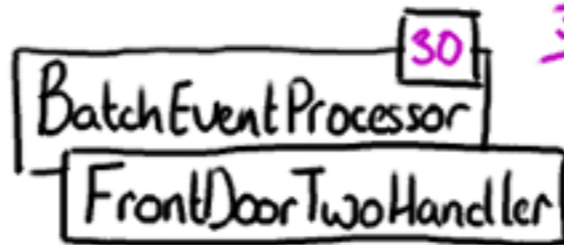
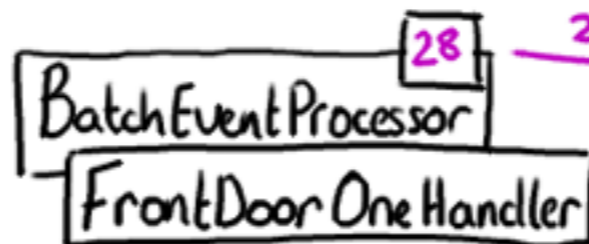
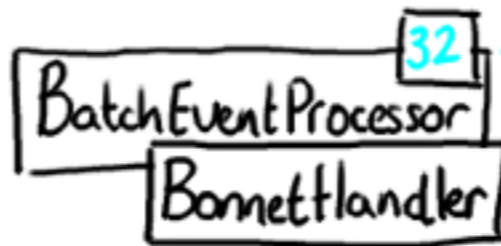
32  
BatchEventProcessor  
RearDoorTwoHandler

Sequence Barrier

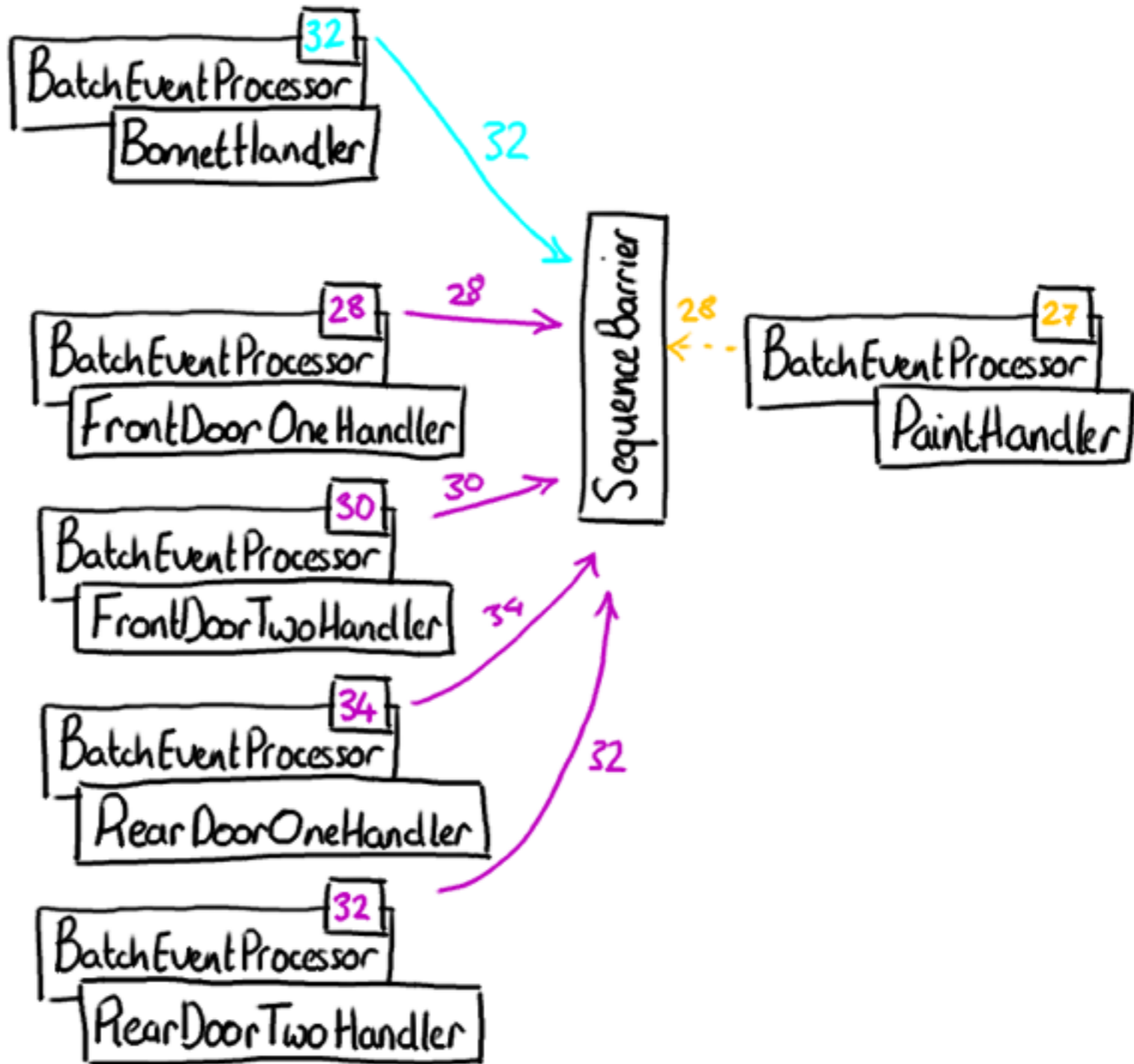
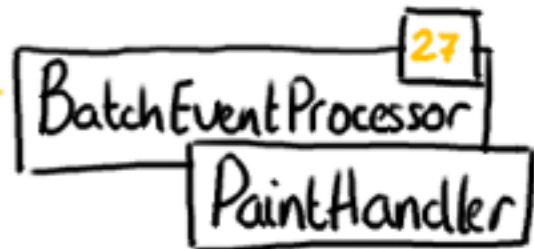


Sequence Barrier

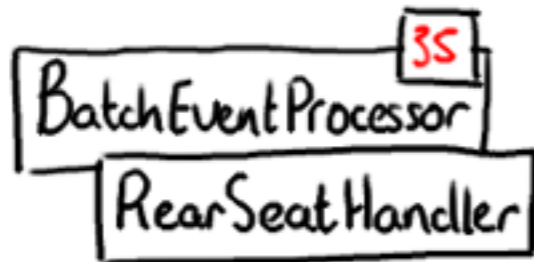
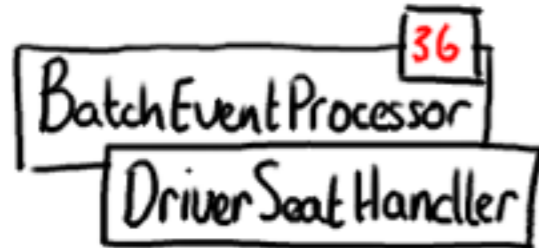
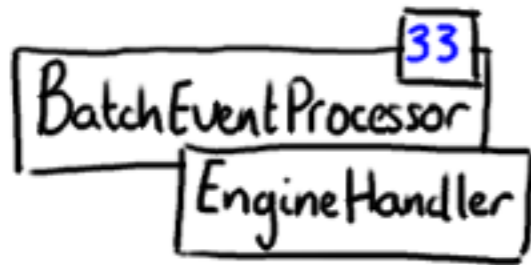
Sequence Barrier



Sequence Barrier

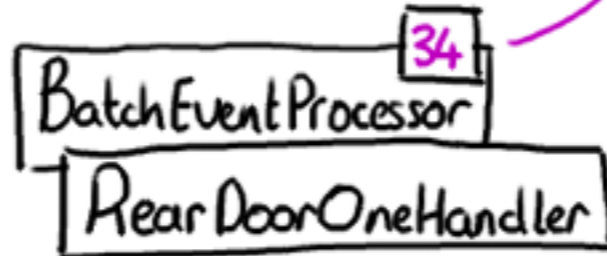
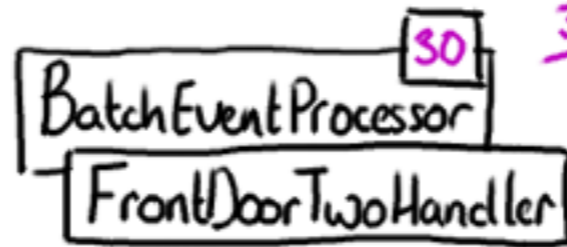
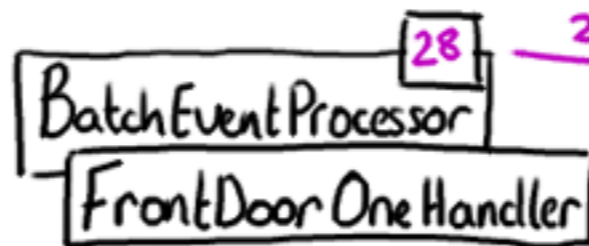
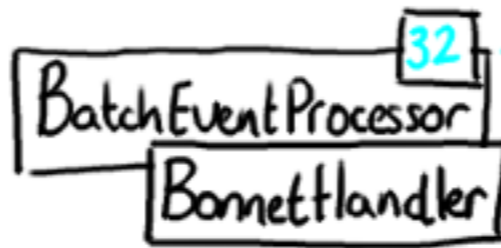


Sequence Barrier

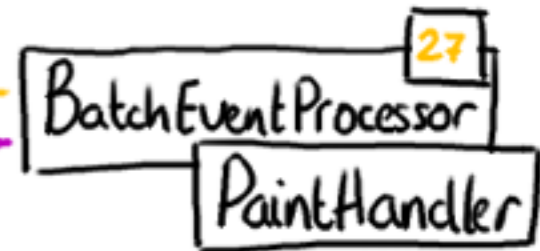


Sequence Barrier

Sequence Barrier



Sequence Barrier



32

28

30

34

32

28

28

27

Sequence Barrier

BatchEventProcessor<sup>33</sup>  
EngineHandler

BatchEventProcessor<sup>36</sup>  
DriverSeatHandler

BatchEventProcessor<sup>37</sup>  
PassengerSeatHandler

BatchEventProcessor<sup>35</sup>  
RearSeatHandler

Sequence Barrier

Sequence Barrier

BatchEventProcessor<sup>32</sup>  
BonnetHandler

BatchEventProcessor<sup>28</sup>  
FrontDoorOneHandler

BatchEventProcessor<sup>30</sup>  
FrontDoorTwoHandler

BatchEventProcessor<sup>34</sup>  
RearDoorOneHandler

BatchEventProcessor<sup>32</sup>  
RearDoorTwoHandler

Sequence Barrier

BatchEventProcessor<sup>27</sup>  
PaintHandler

Sequence Barrier

BatchEventProcessor<sup>27</sup>  
WheelOneHandler

BatchEventProcessor<sup>27</sup>  
WheelTwoHandler

BatchEventProcessor<sup>27</sup>  
WheelThreeHandler

BatchEventProcessor<sup>27</sup>  
WheelFourHandler



Sequence Barrier

BatchEventProcessor 33  
EngineHandler

BatchEventProcessor 36  
DriverSeatHandler

BatchEventProcessor 37  
PassengerSeatHandler

BatchEventProcessor 35  
RearSeatHandler

Sequence Barrier

Sequence Barrier

BatchEventProcessor 32  
BonnetHandler

BatchEventProcessor 28  
FrontDoorOneHandler

BatchEventProcessor 30  
FrontDoorTwoHandler

BatchEventProcessor 34  
RearDoorOneHandler

BatchEventProcessor 32  
RearDoorTwoHandler

Sequence Barrier

BatchEventProcessor 27  
PaintHandler

Sequence Barrier

BatchEventProcessor 27  
WheelOneHandler

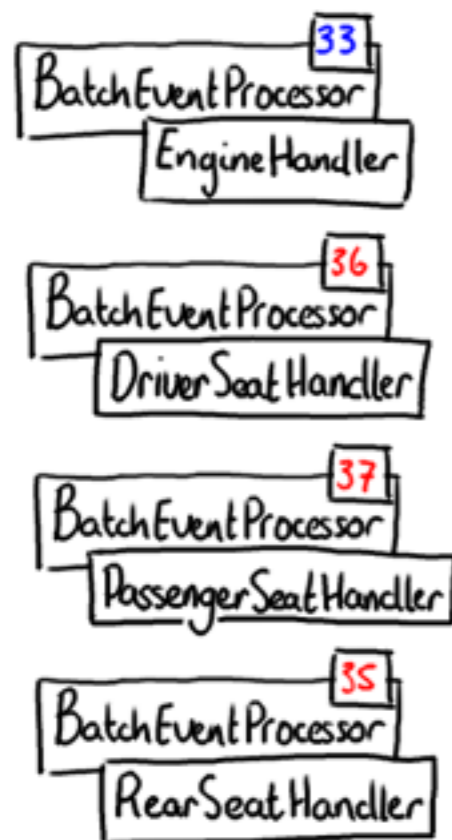
BatchEventProcessor 27  
WheelTwoHandler

BatchEventProcessor 27  
WheelThreeHandler

BatchEventProcessor 27  
WheelFourHandler

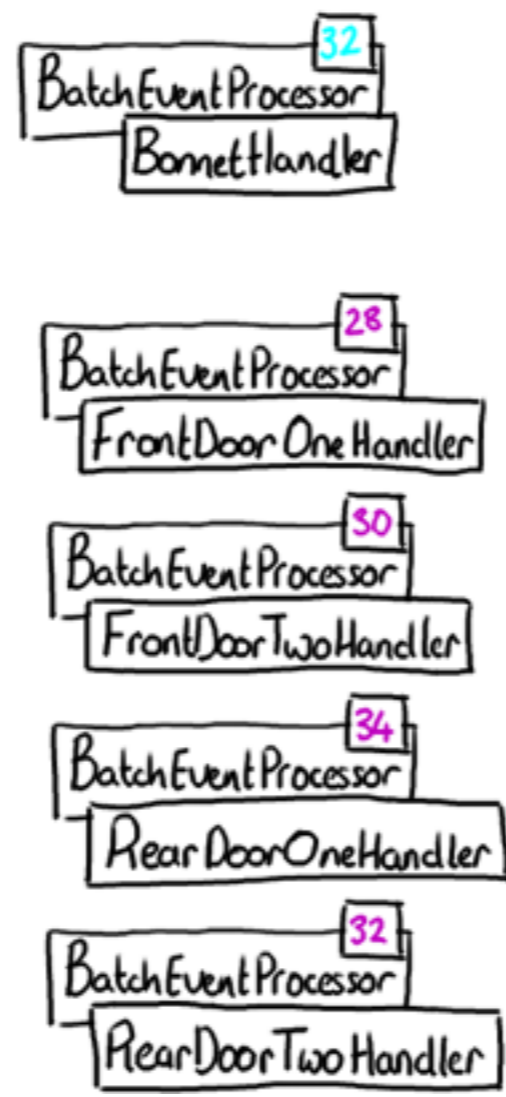


Sequence Barrier



Sequence Barrier

Sequence Barrier

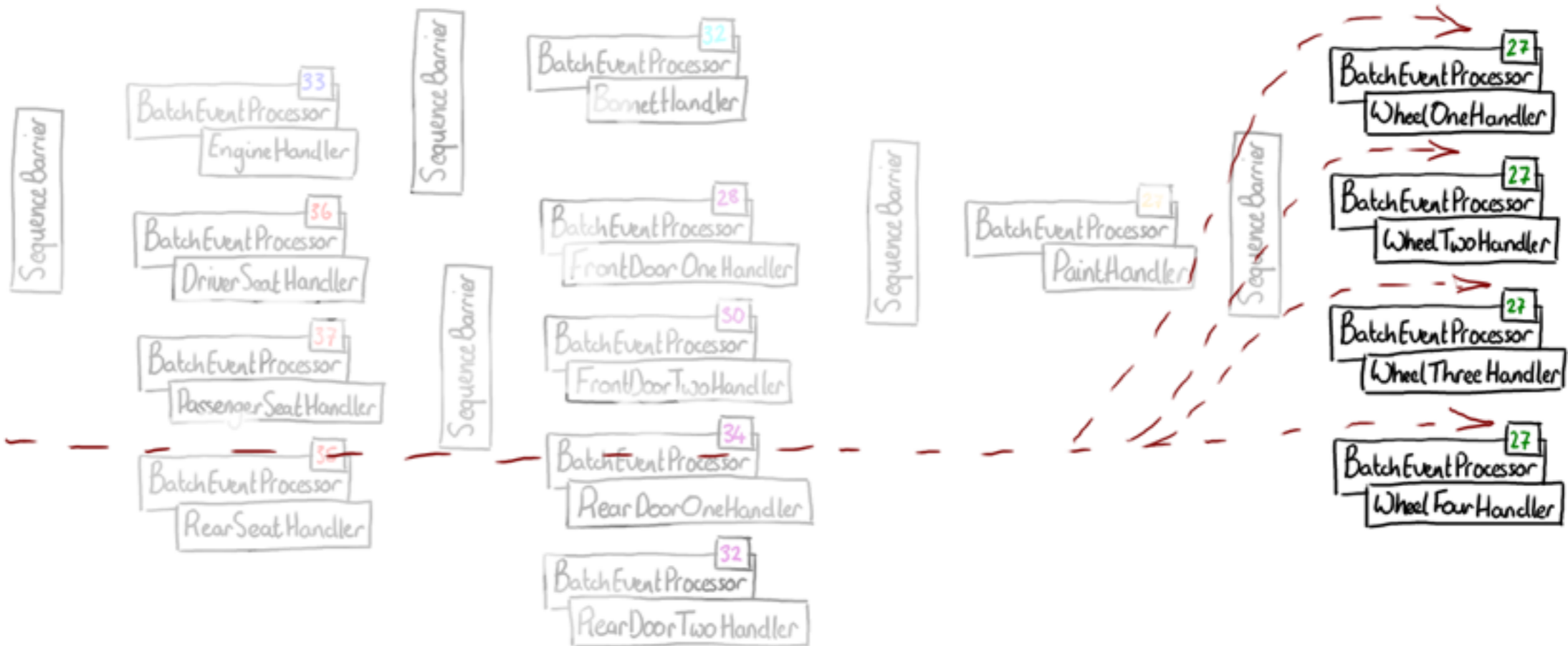
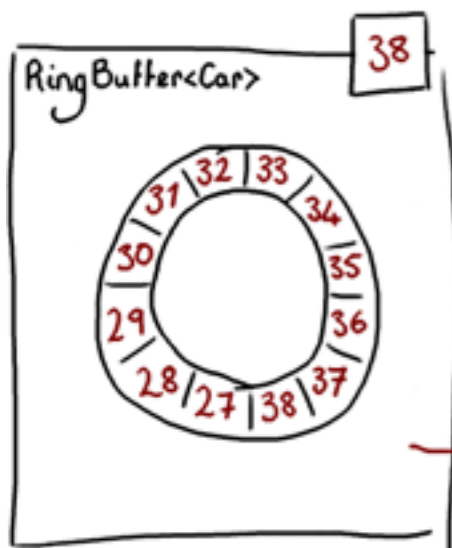


Sequence Barrier



Sequence Barrier





# Don't wrap the buffer!

```
ringBuffer.setGatingSequences(finalEventProcessor.getSequence());
```

# Caveats

# Is that it?

- Wait and claim strategies
- Multiple publishers
- Different EventHandlers
- The Wizard

# You get...

- A framework that encourages you to model your domain
- The ability to run in parallel but single-threaded
- Reliable ordering
- ...and it can be very fast

# More Information

- Google Code Site, including Wiki  
<http://code.google.com/p/disruptor/>
- Blogs, e.g. mine: [mechanitis.blogspot.com](http://mechanitis.blogspot.com)
- Presentations
- Google Group



# Q&A

- We are hiring!
- [careers@lmax.com](mailto:careers@lmax.com)