Ruby Threads
(and so can you!)

Johnny Shields
Founder & CTO, TableCheck

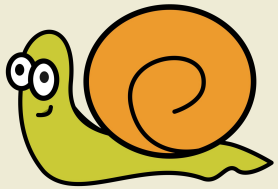
TableCheck

CYBERGIZER

Example Problem:

A Rake task which sends lots of emails

```ruby
desc "Send lots of emails"
task send_emails: :environment do
  Customer.find_each do |c|
    MyMailer.with(customer: c).send
  end
end
```

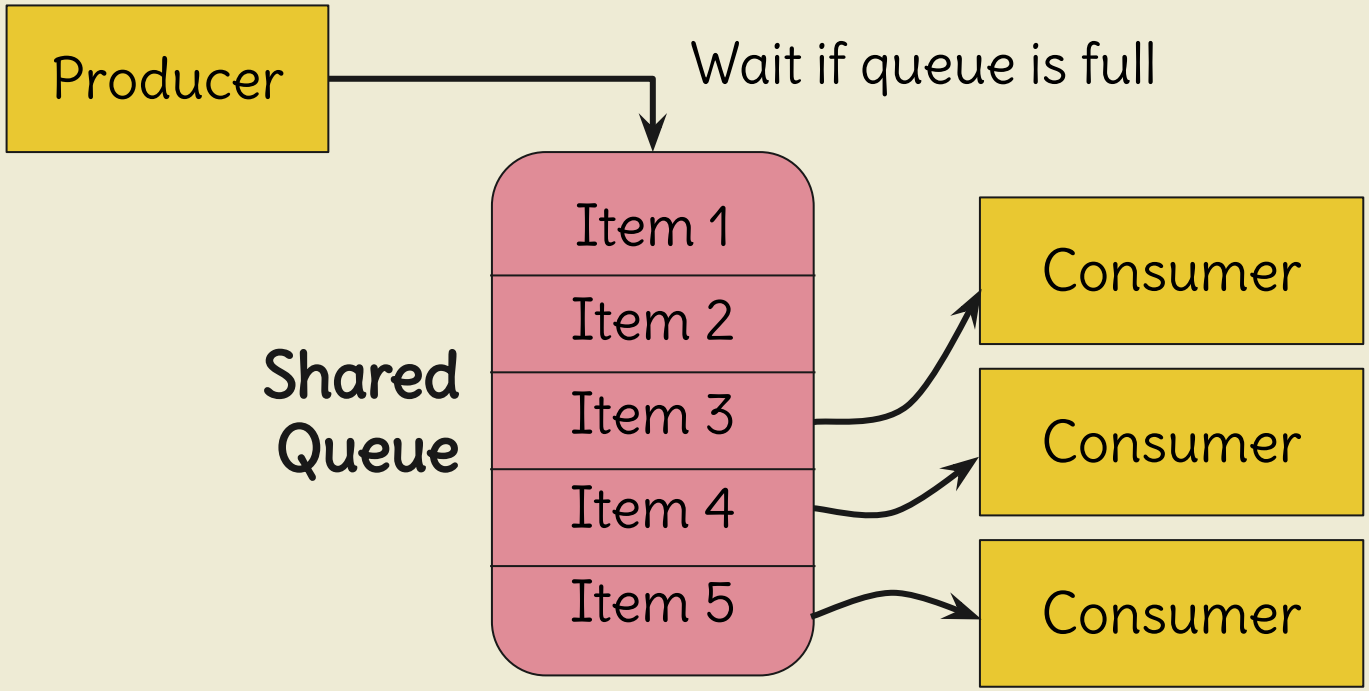


Synchronous mailer I/O is slooowww...

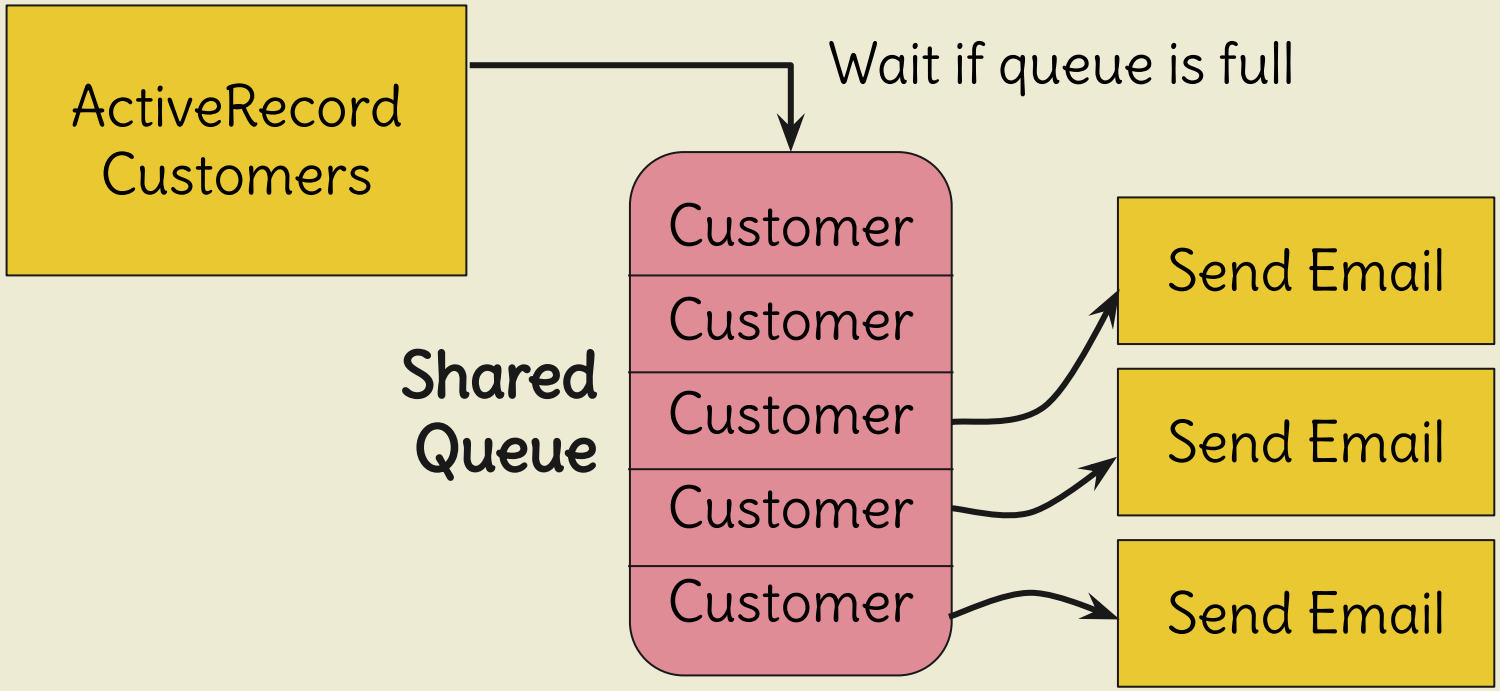

GOTTA GO
FAST

The Producer Consumer Pattern

The Producer-Consumer Pattern



Producer-Consumer: Sending Mails



```ruby
THREADS = 50

def process
  @queue = SizedQueue.new(THREADS * 2)

  _producer = make_producer_thread
  consumers = Array.new(THREADS) { make_consumer_thread }
  consumers.each(&:join)
end
```

```ruby
def make_producer_thread
  Thread.new do
    Customer.find_each do |customer|
      @queue << customer
    end
  ensure
    THREADS.times { @queue << :eoq } # end object
  end
end

def make_consumer_thread
  Thread.new do
    until (customer = @queue.pop) == :eoq
      MyMailer.with(customer: customer).send
    end
  end
end
```

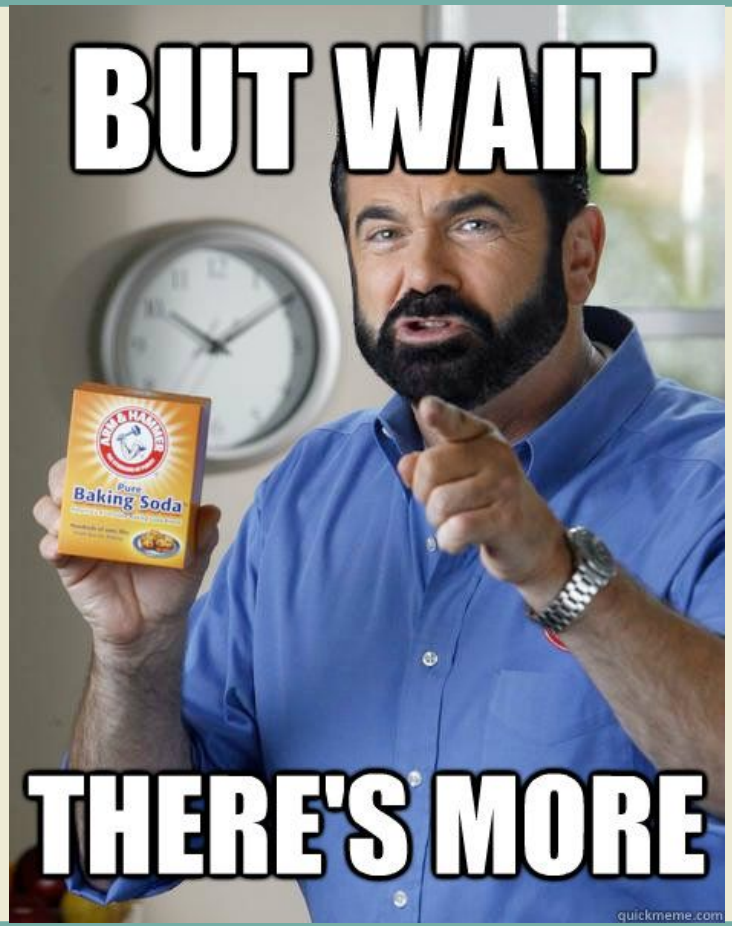
OK Let's benchmark!

```ruby
require 'benchmark'

# 1000 customers
# 10 milliseconds per mail
Benchmark.bm do |bm|
  bm.report('single thread') do
    Customer.find_each { |c| MyMailer.with(customer: c).send }
  end

  bm.report('producer consumer - 50 threads') do
    ProducerConsumer.new.process
  end
end
```

```
                  user     system      total        real
single thread  0.000000   0.000000   0.000000 ( 15.618268)
producer consumer - 50 threads  0.016000   0.031000   0.047000 (  0.419344)
```

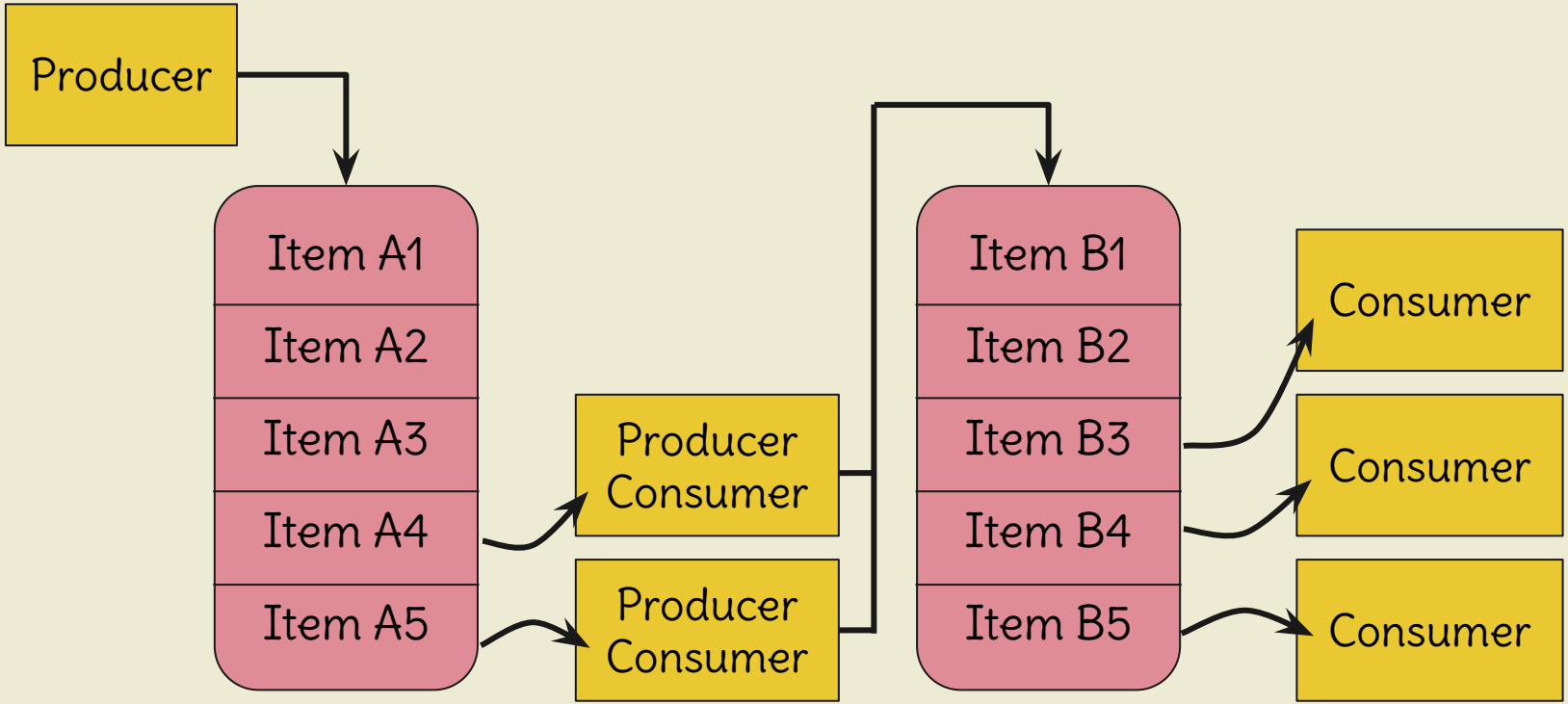

BUT WAIT
Pure
Baking Soda
THERE'S MORE
quickmeme.com

The Multi-Stage Pattern

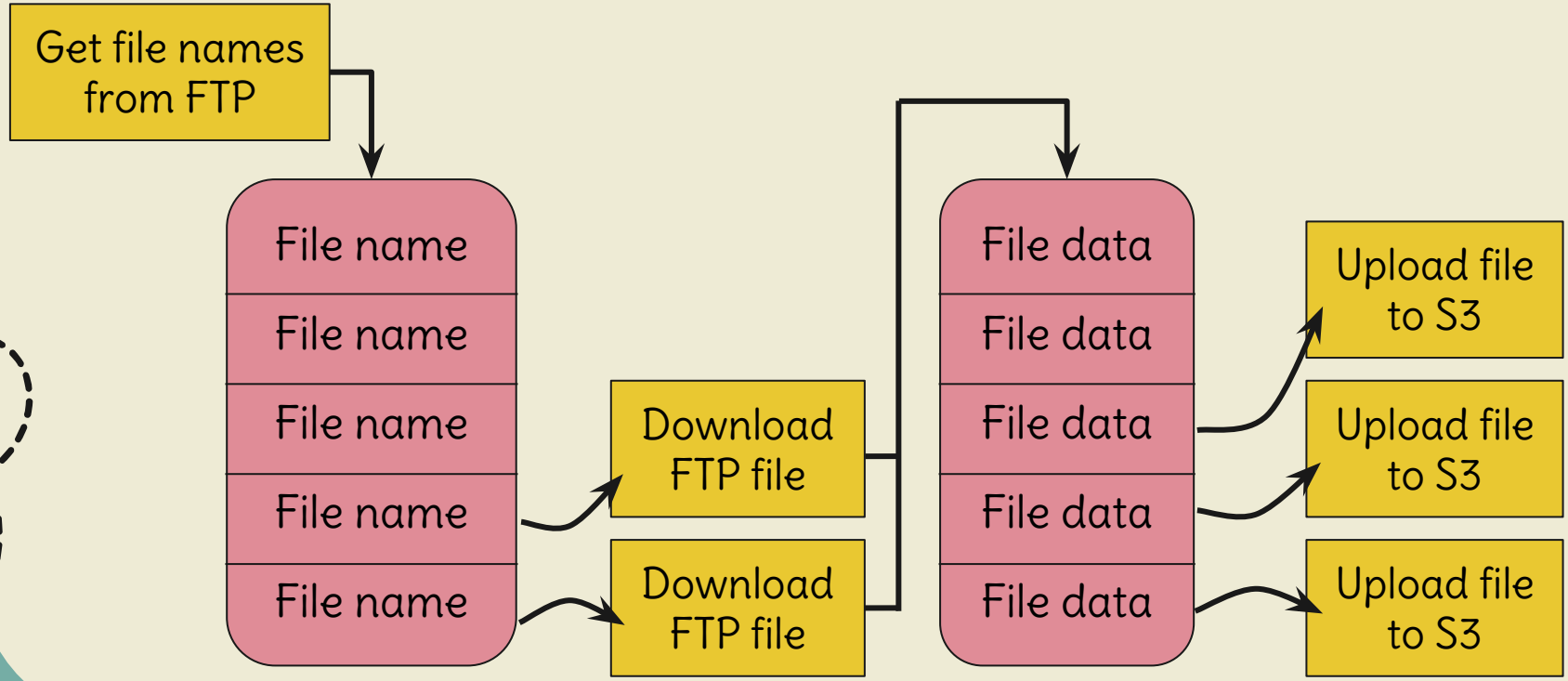
Example Problem:

Download files from FTP and upload to Amazon S3

The Multi-Stage Pattern



Multi-Stage: Move files from FTP to S3



```ruby
THREADS = 5

def process
  @download_queue = SizedQueue.new(THREADS * 2)
  @upload_queue = SizedQueue.new(THREADS * 2)

  _producer = make_download_producer
  producer_consumer = make_upload_producer
  consumers = make_upload_consumers

  producer_consumer.join
  consumers.each(&:join)
end
```

```ruby
def make_download_producer
  Thread.new do
    MyFtp.get_file_names('/my/directory').each do |file_name|
      @download_queue << file_name
    end
  ensure
    THREADS.times { @download_queue << :eoq }
  end
end
```

```ruby
# includes download consumers!
def make_upload_producer
  Thread.new do
    download_consumers = Array.new(THREADS) do
      Thread.new do
        until (file_name = @download_queue.pop) == :eoq
          @upload_queue << MyFtp.download_file(file_name)
        end
      end
    end
    download_consumers.each(&:join)
  ensure
    THREADS.times { @upload_queue << :eoq }
  end
end
```

```ruby
def make_upload_consumers
  Array.new(THREADS) do
    Thread.new do
      until (file_data = @upload_queue.pop) == :eoq
        Aws::S3.upload("my/path/#{file_data.name}", file_data)
      end
    end
  end
end
```

It's benchmark time (again)!!

```ruby
require 'benchmark'

# 1000 files
# 10 milliseconds per upload/download
Benchmark.bm do |bm|
  bm.report('single thread') do
    MyFtp.get_file_names('my/directory').each do |file_name|
      file_data = MyFtp.download_file(file_name)
      Aws::S3.upload(file_data)
    end
  end

  bm.report('producer consumer - 50 threads') do
    MultiStage.new.process
  end
end
```

```
                user     system      total        real
single thread  0.000000   0.000000   0.000000 ( 31.247323)
producer consumer - 50 threads  0.063000   0.141000   0.204000 (  0.582674)
```

If you want to do it better...

Check out Elixir's GenStage

Thanks!

Get the code

https://github.com/TableCheck-Labs/euruko-2023-threads

TableCheck is hiring!

https://careers.tablecheck.com/

TableCheck

CYBERGIZER