

JBoss Community

TorqueBox

Ruby na sterydach

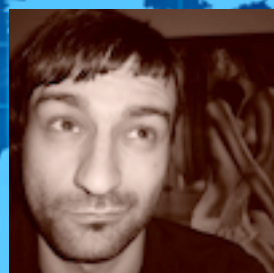
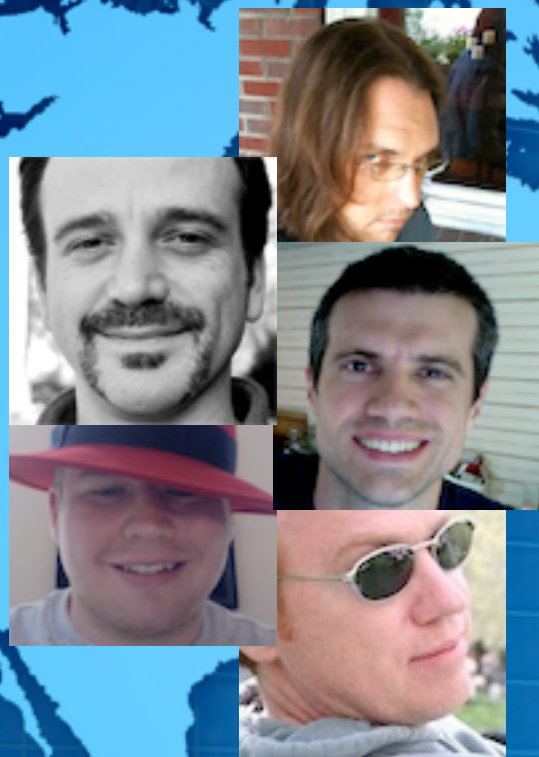


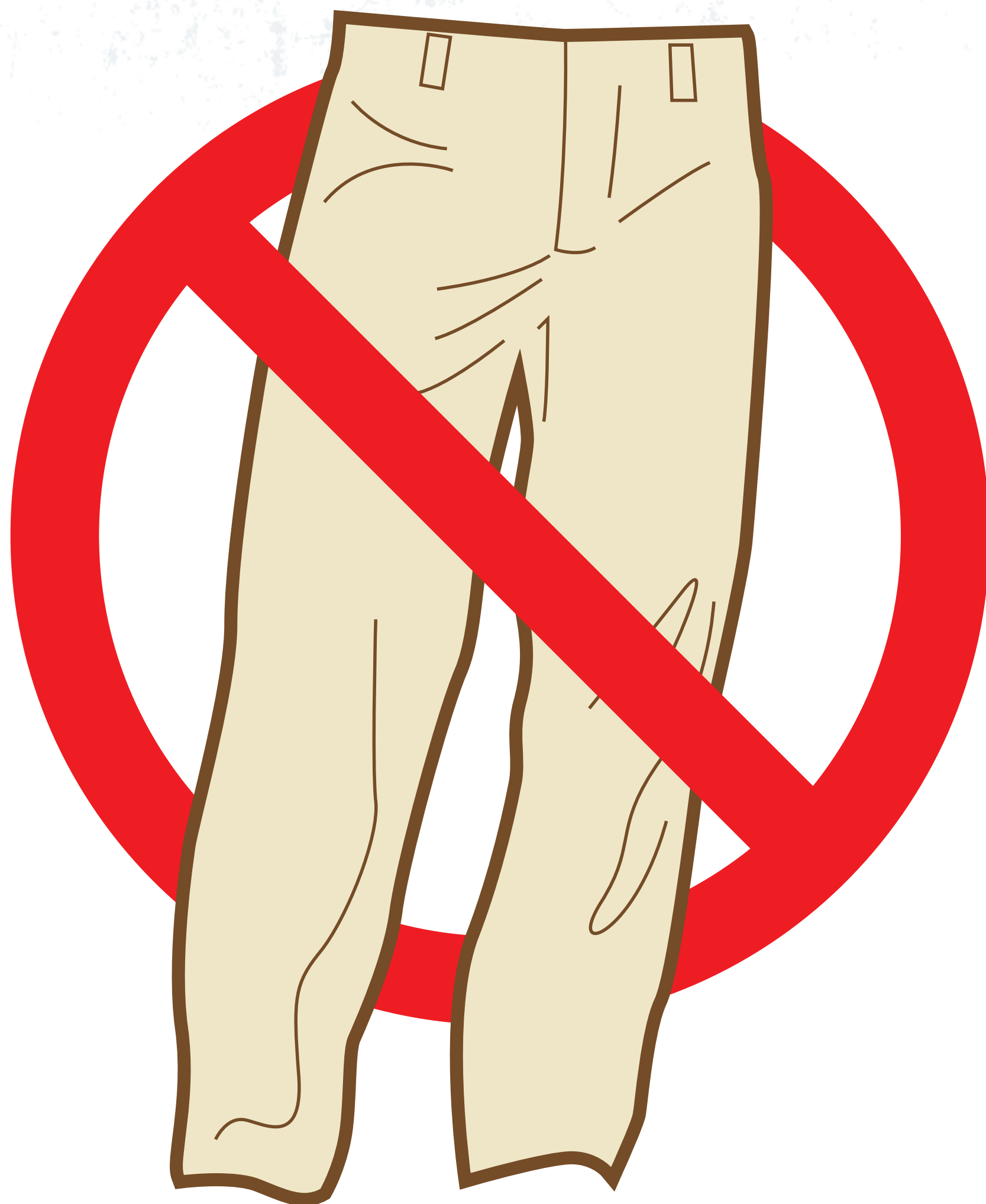
Marek.



JBoss Community

project:odd







BoxGrinder



JBoss Application Server 7

+

fedora^f

=



JBoss Community



**TorqueBox to klej
pomiedzy JRuby a
serwerem aplikacji
JBoss AS.**

JRuby.

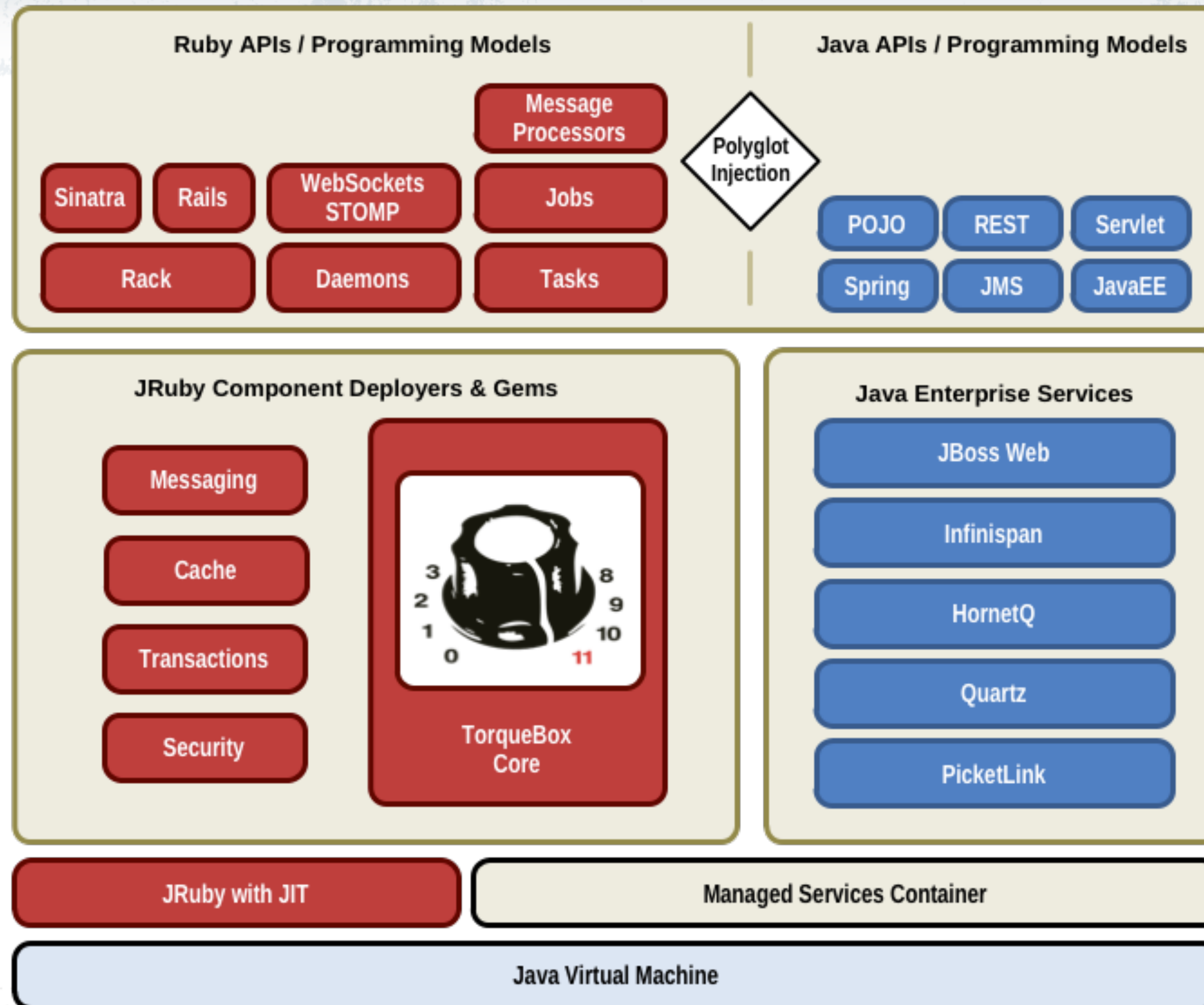
Prawdziwe wątki.

**Bardzo szybkie
środowisko.**

Biblioteki Javy.

JBoss AS.

Serwer aplikacji Java EE (klastrowanie, web, wiadomości, cache, etc)



Cel.

Pierwszorzędny

Rails / Sinatra / Rack

Drugorzędny

Usługi / Zadania /
Wiadomości / Transakcje /
Cache / WebSockets / HA

Web.

Sinatra

```
myapp/  
  config.ru  
  myapp.rb  
  torquebox.yml
```

```
myapp/  
  config/  
    torquebox.yml  
    application.rb  
    ...  
  app/  
    views/  
    controllers/  
    models/  
    ...
```

Rails

Poka poka.

Nuda.

Usługi.

start()

...czas mija...

stop()

Gdy wykonujemy
coś w tle przez
jakiś czas, np.
pobieranie danych z
Twittera

```
class MyService
  def initialize(opts)
  end

  def start
  end

  def stop
  end
end
```

app/services/my_service.rb

```
services:  
  MyService:  
    config:  
      foo: bar  
      abc: 1  
  
  MySingletonService:  
    singleton: true
```

config/torquebox.yml

singleton: true

**Jedna aktywna usługa w
jednym czasie w klastrze.**

singleton: false

Usługi aktywne na
wszystkich węzłach
klastra (domyślnie).

Zadania.

Gdy wykonujemy
coś co jakiś czas.

Co piątek
W pierwszy dzień miesiąca
Co 2 minuty

No, wiesz o co chodzi.

```
class MyJob  
  def run  
  end  
end
```

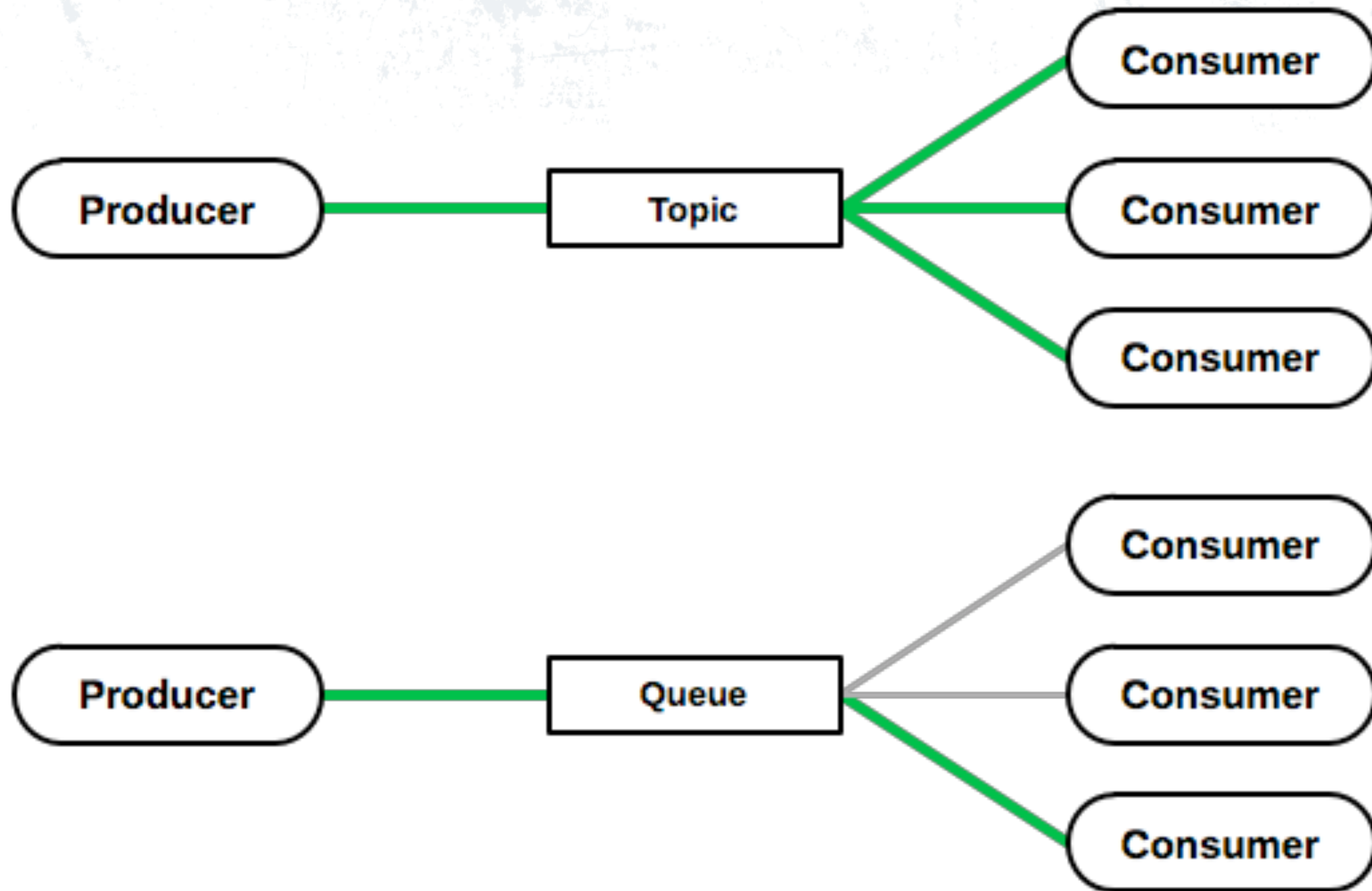
app/jobs/my_job.rb

```
jobs:  
  newsletter:  
    job: MyJob  
    cron: '0 0 0 1 * ?'  
    singleton: true  
  
cache_cleaner:  
  job: CacheCleanerJob  
  cron: '5 * * * * ?'
```

config/torquebox.yml

Wiadomości.

JMS.



```
queues:  
  /queues/foo:  
  /queues/bar:
```

```
topics:  
  /topics/foo:  
  /topics/bar:
```

config/torquebox.yml

Wysyłamy...

```
class MyController < ApplicationController
  include TorqueBox::Injectors

  def index
    queue = inject( '/queues/foo' )
    queue.publish( "Lecimy!" )
  end

end
```

**app/controllers/
my_controller.rb**

...i odbieramy

```
include TorqueBox::Messaging
```

```
class MyProcessor < MessageProcessor
  def initialize(opts)
    end

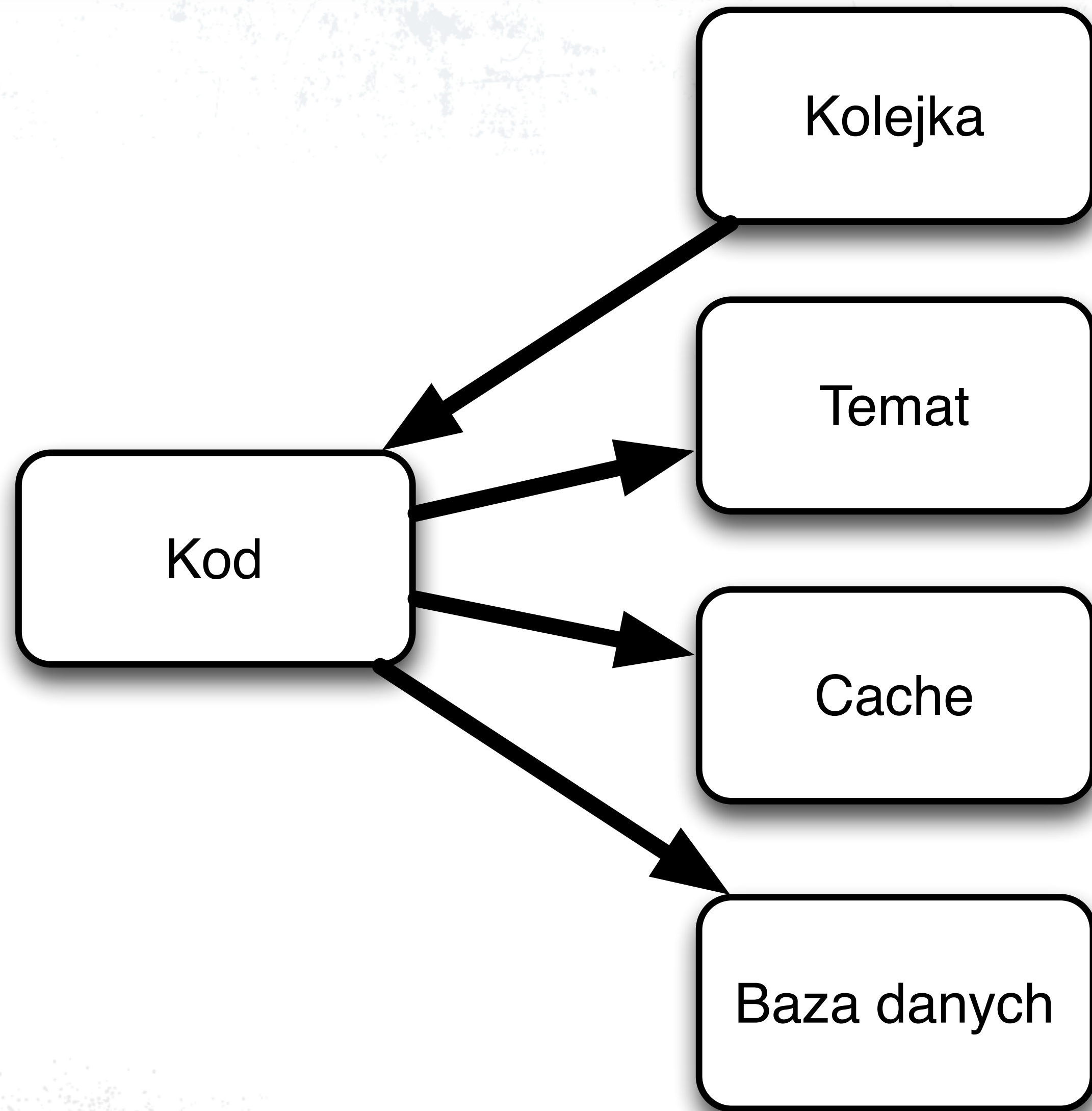
    def on_message(body)
      # body == "Lecimy!"
    end
  end
end
```

**app/processors/
my_processor.rb**

```
messaging:  
  /queues/foo:  
    MyProcessor:  
      filter: "language = 'ruby'"  
      concurrency: 3  
      config:  
        foo: bar
```

config/torquebox.yml

Transakcje.



Dystrybuowane.

(trudne słowo)

Sukces tylko wtedy gdy
wszystko się powiedzie.

Transakcje w AR?

```
TorqueBox.transaction do
  User.create(:username => 'Kotori')
  TorqueBox.transaction do
    User.create(:username => 'Nemu')
    raise ActiveRecord::Rollback
  end
end
```

Transakcje gdziekolwiek?

```
TorqueBox.transaction do  
  inject( '/queues/foo' ).publish("blah")  
end
```

Cache.

Infinispan.

Data Grid. Replikowalna,
dystribuowalna pamięć.

Zamiast memcached.

```
cache = TorqueBoxStore.new(  
  :name => 'foo',  
  :mode => :replicated  
)
```

```
cache.put(...)  
cache.get(...)
```

Wstrzykiwanie zasobów.

Usługi Kolejki / tematy Java CDI

```
inject( '/queues/foo' ).publish(...)  
inject( '/topics/bar' ).publish(...)
```

CDI

```
package com.acme.blah;  
  
@ApplicationScoped  
public class MyBean {  
    public void fire() { ... }  
}
```

```
bean = inject(com.acme.blah.MyBean)  
bean.fire
```

Klastrowanie.

**JBoss AS ma wbudowane
wspacie dla klastrowania.**

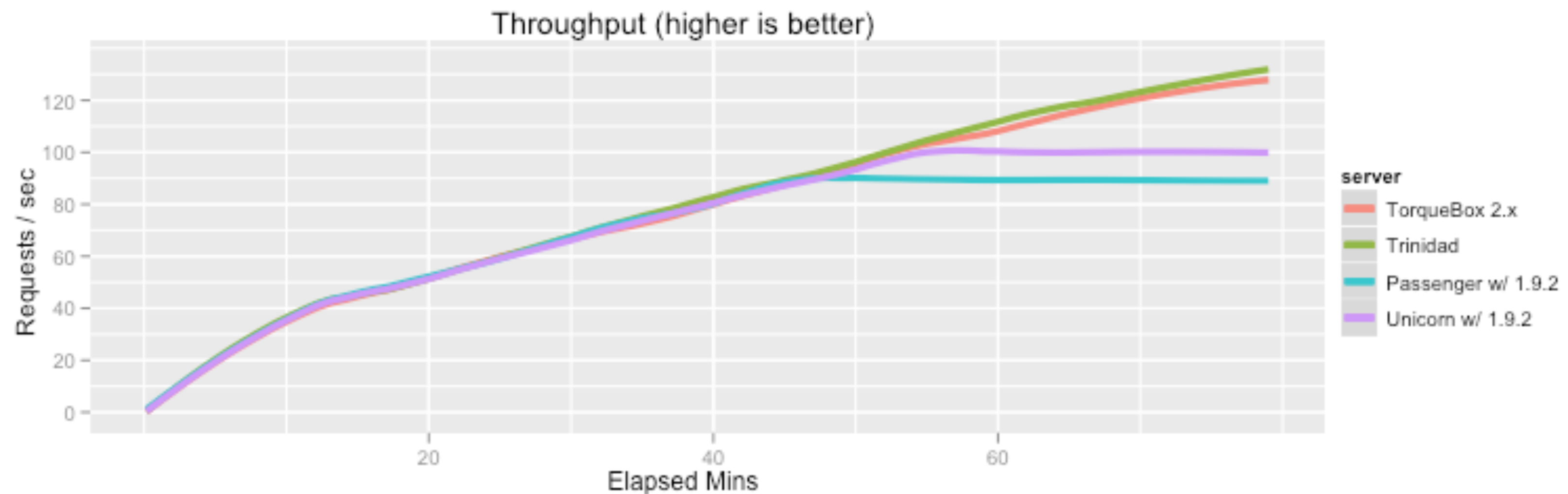
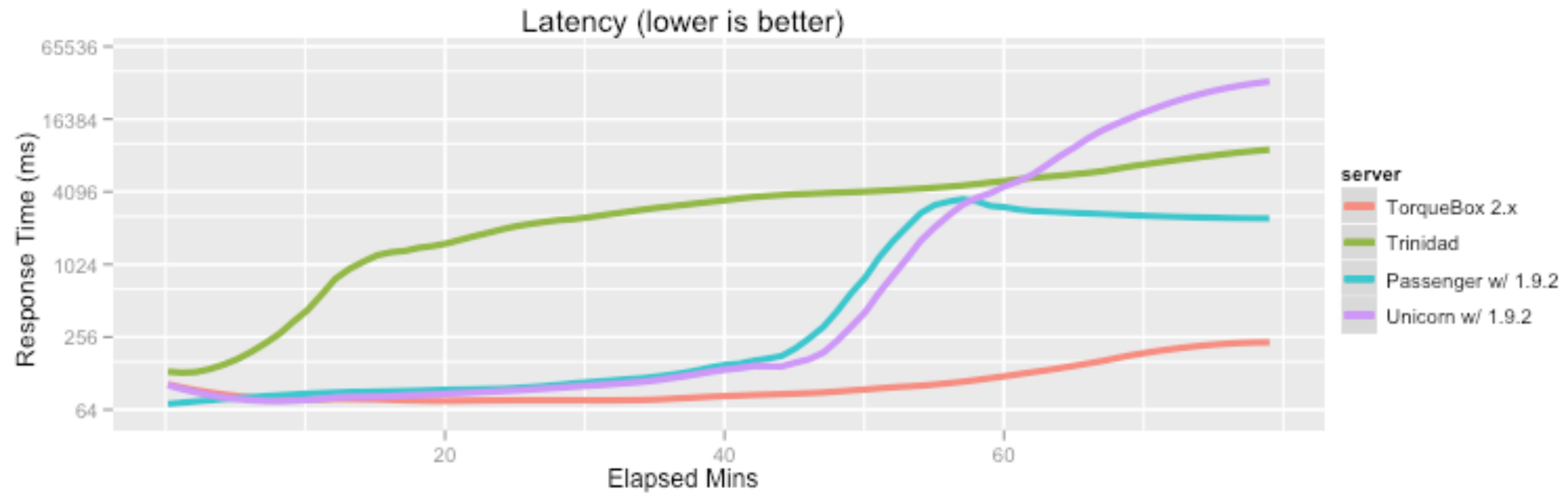
mod_cluster

**Jeszcze nie
wspomniałem o...**

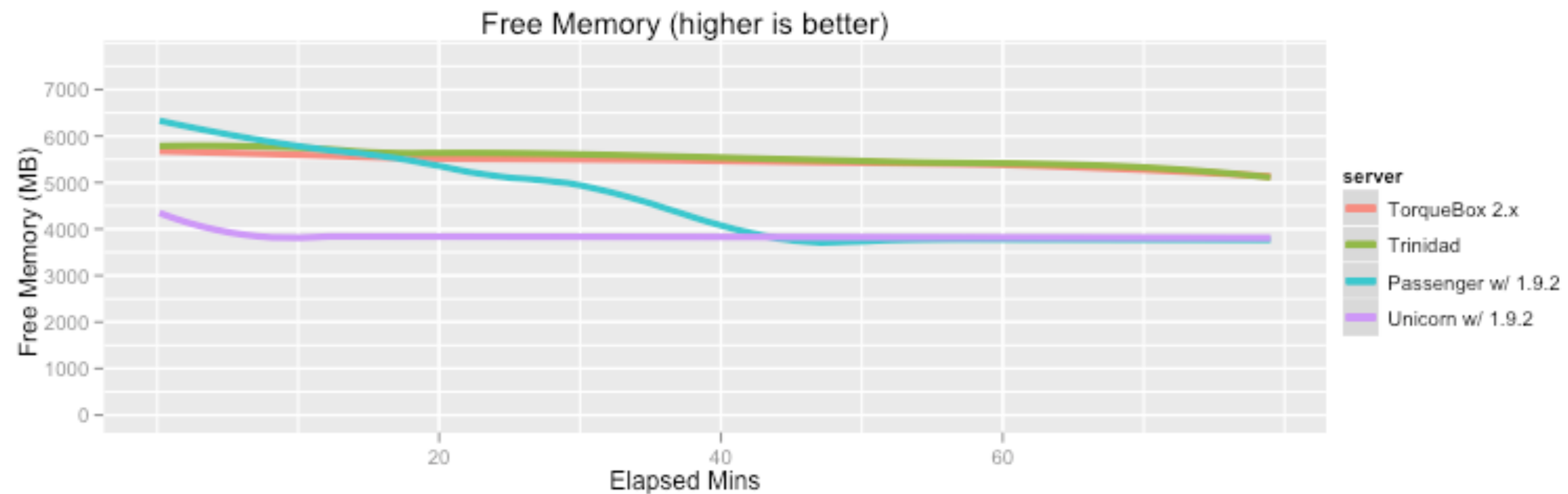
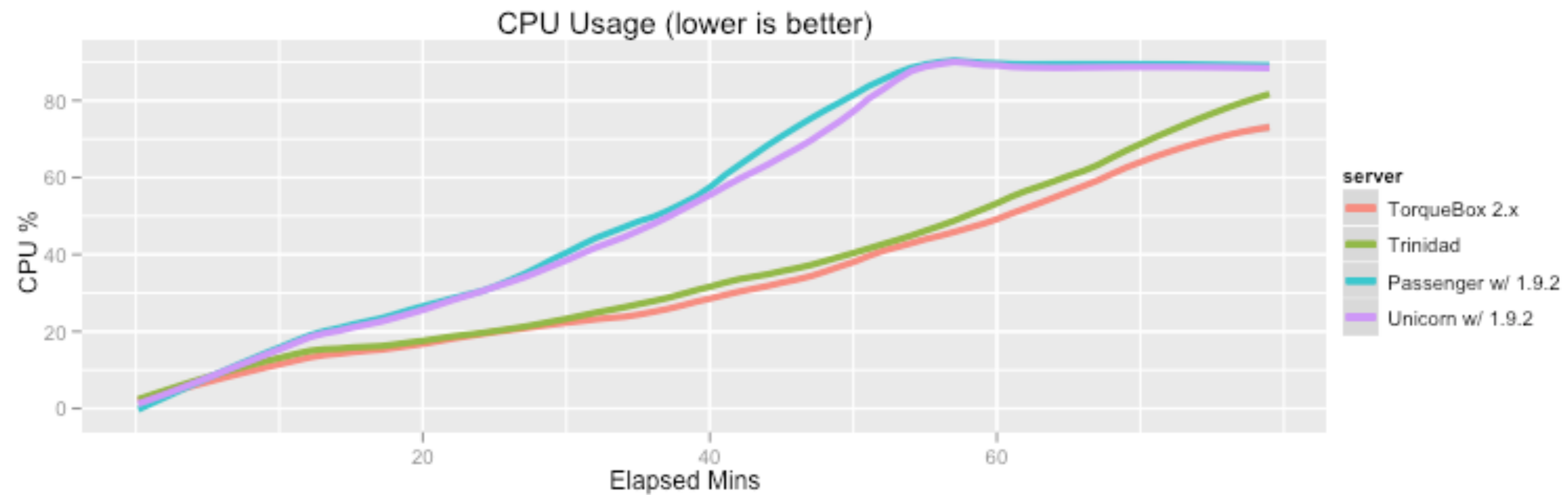
Backgroundables.

WebSockets.

Kilka słów o wydajności.



<http://torquebox.org/news/2011/10/06/torquebox-2x-performance/>



<http://torquebox.org/news/2011/10/06/torquebox-2x-performance/>

<http://torquebox.org/>
<http://github.com/torquebox>

IRC: **#torquebox** on FreeNode

@torquebox



Dzięki.

**Nie zapomnij o
naklejkach!**