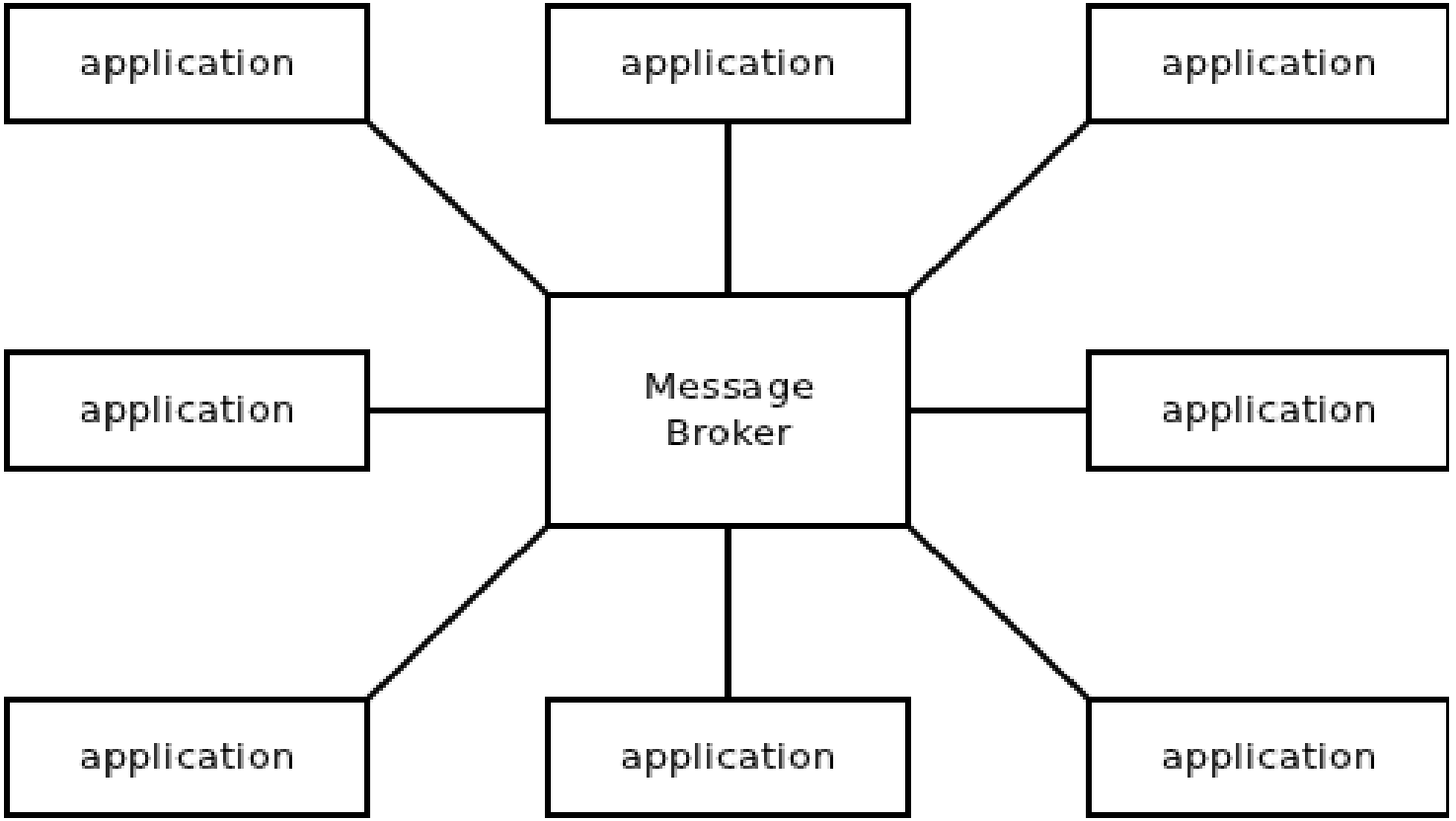
Distributed Messaging
The Administrative Aspect

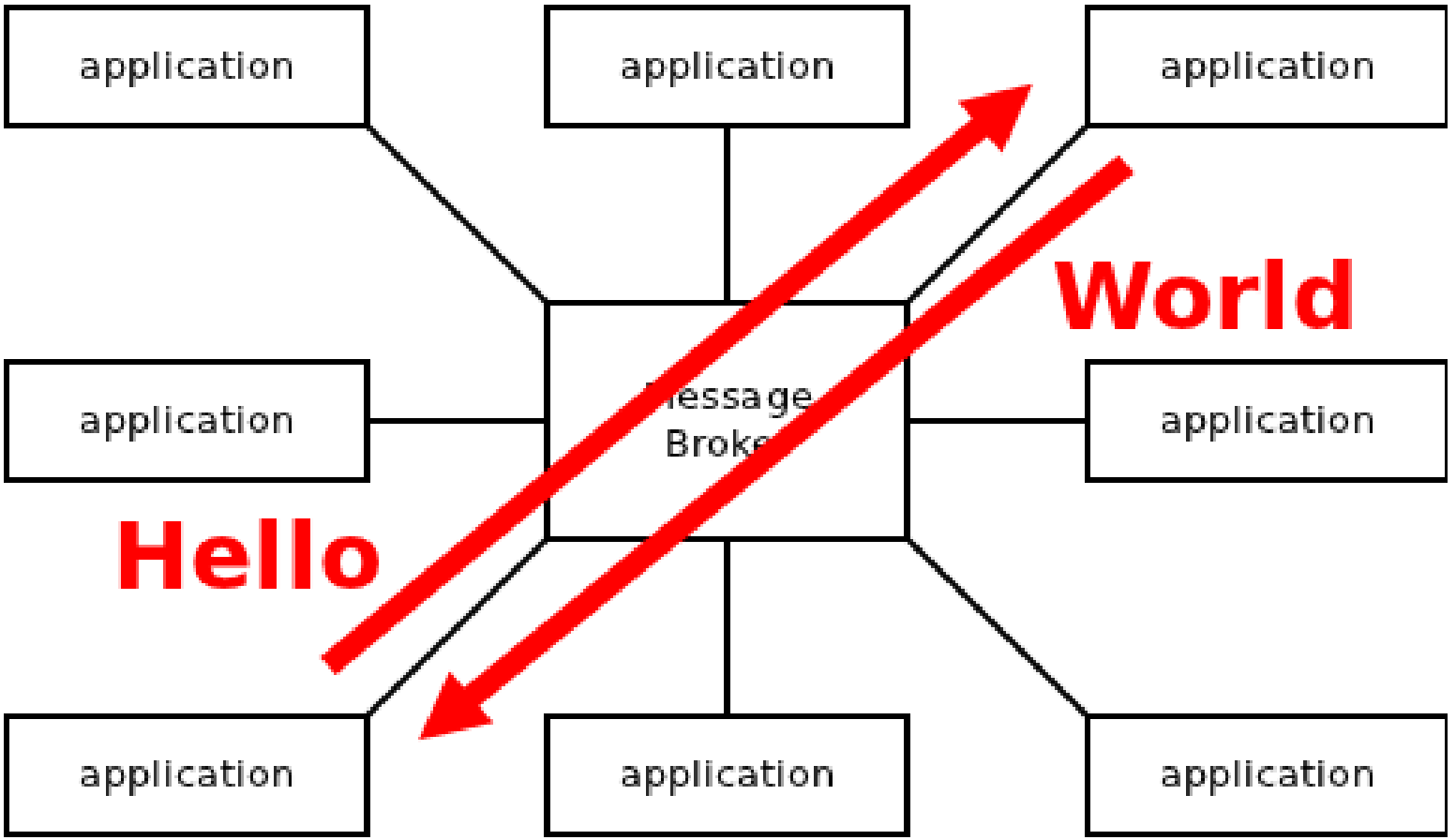
Martin Sústrik
www.250bpm.com
sustrik@250bpm.com

Distributed Messaging

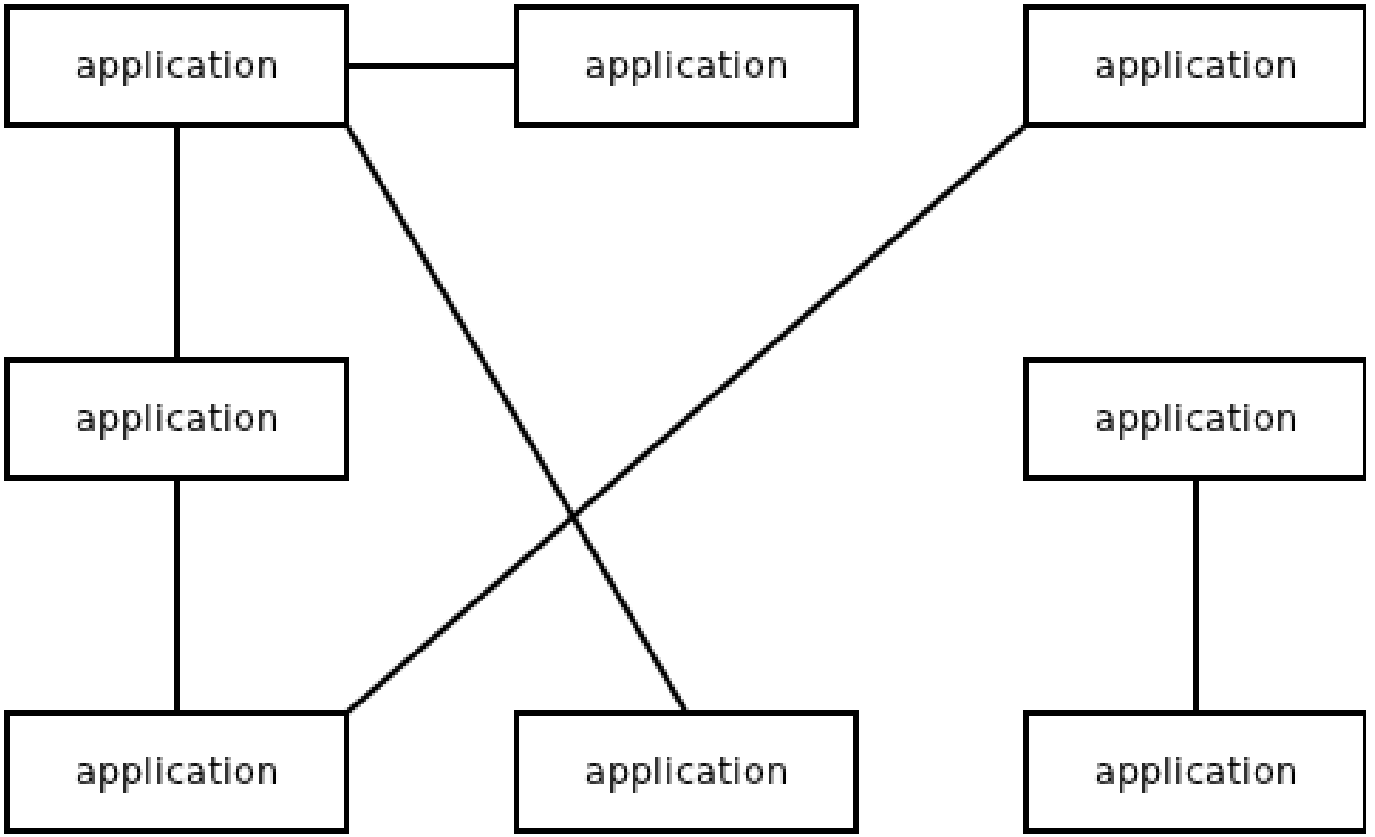
Centralised Messaging



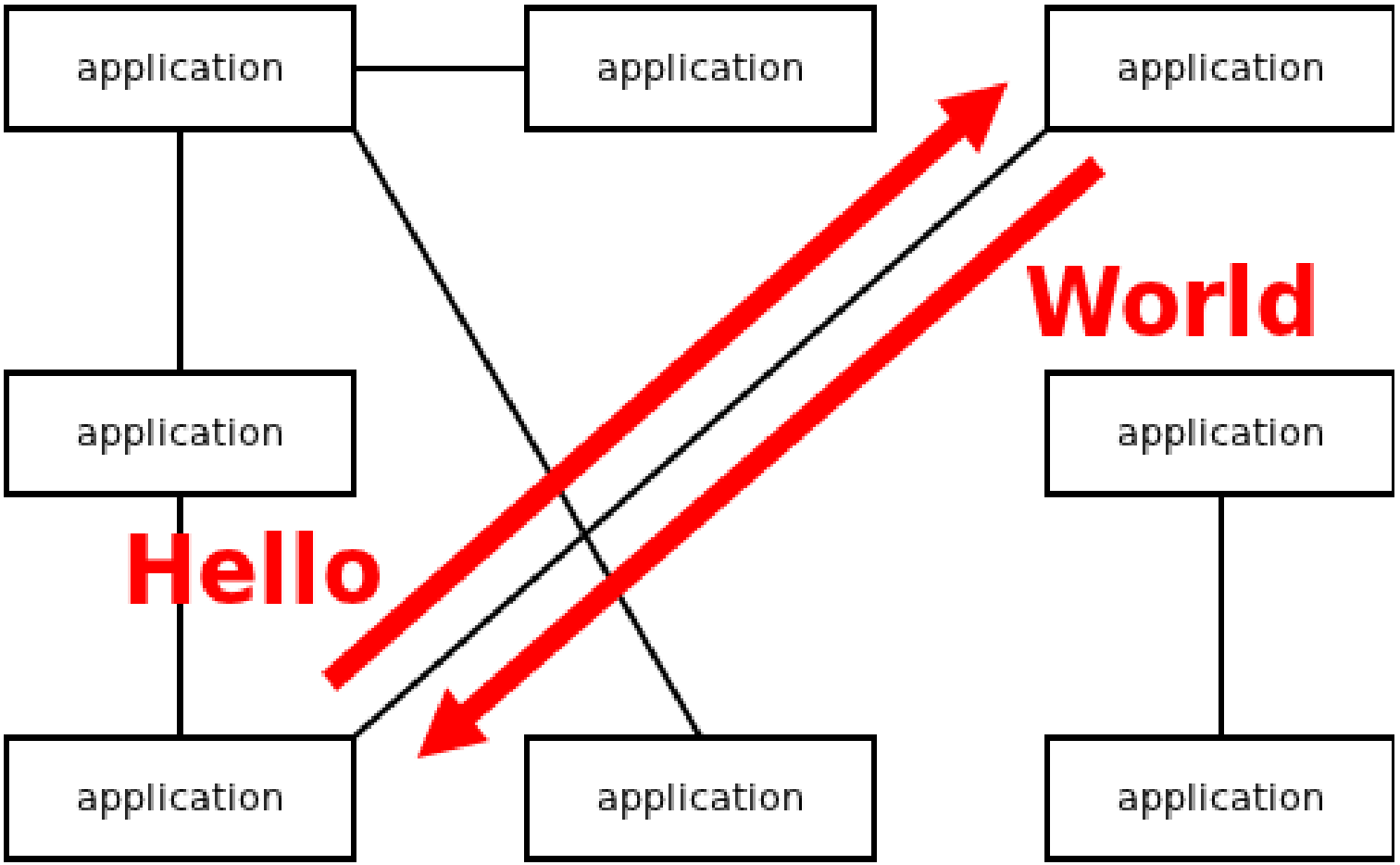
Centralised Messaging



Distributed Messaging

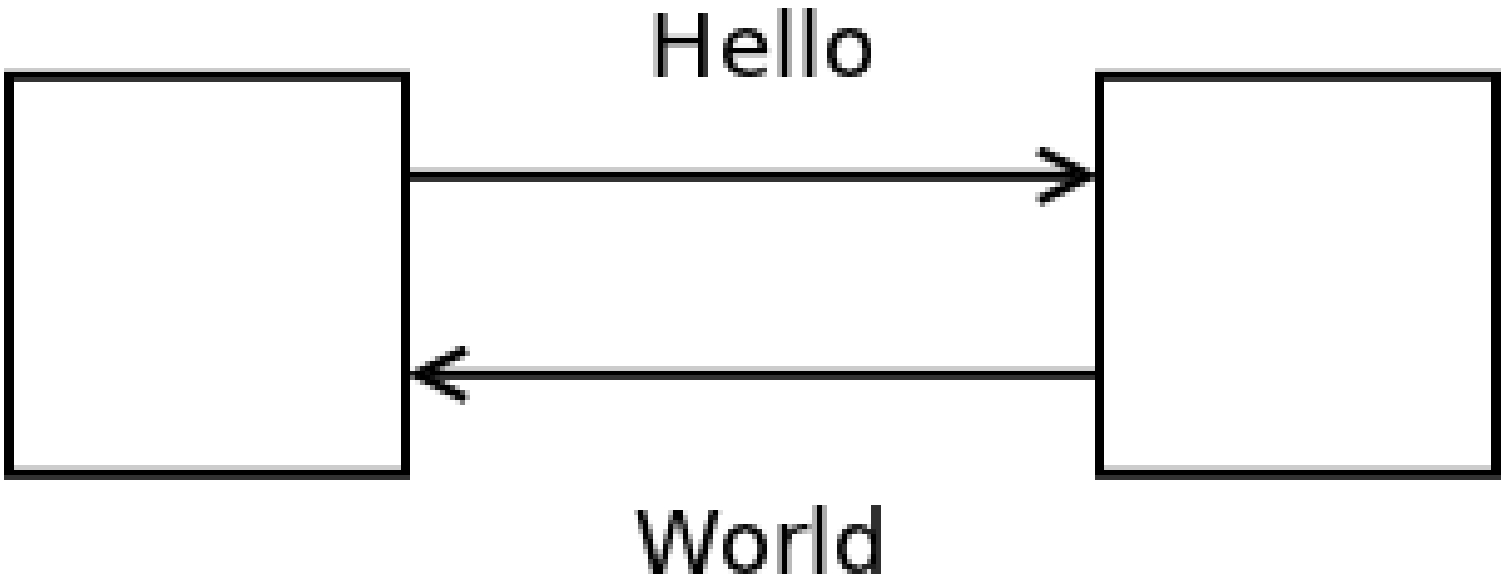


Distributed Messaging



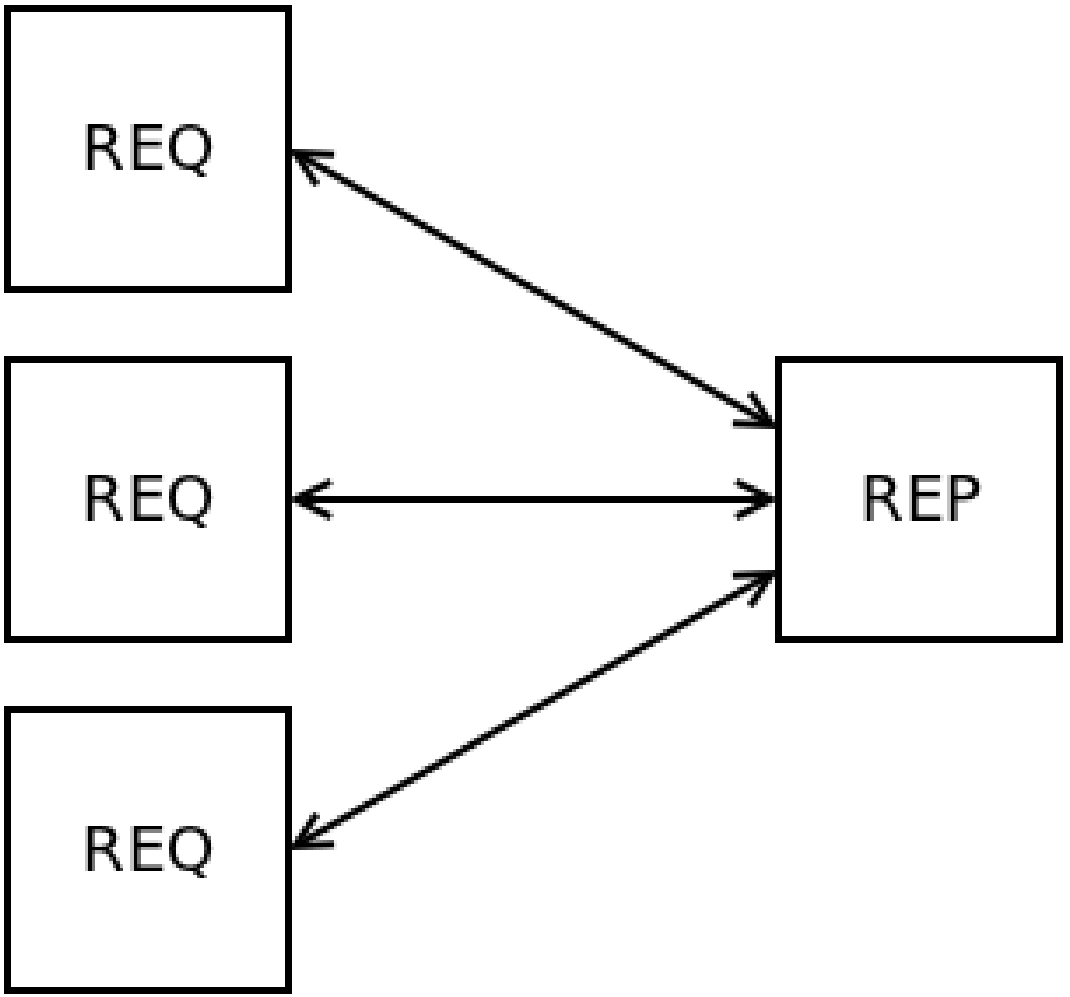
Distributed Messaging Implementation

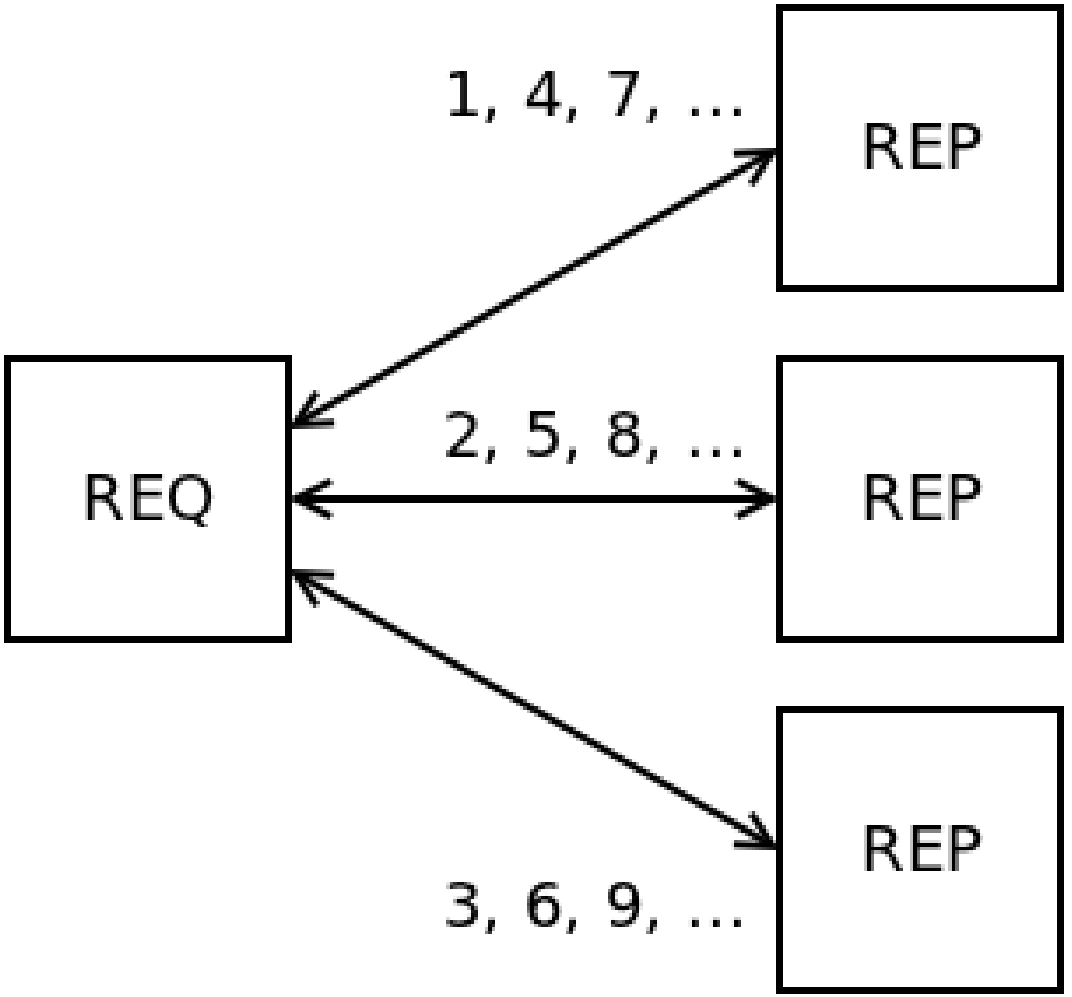
www.zeromq.org


Hello
World

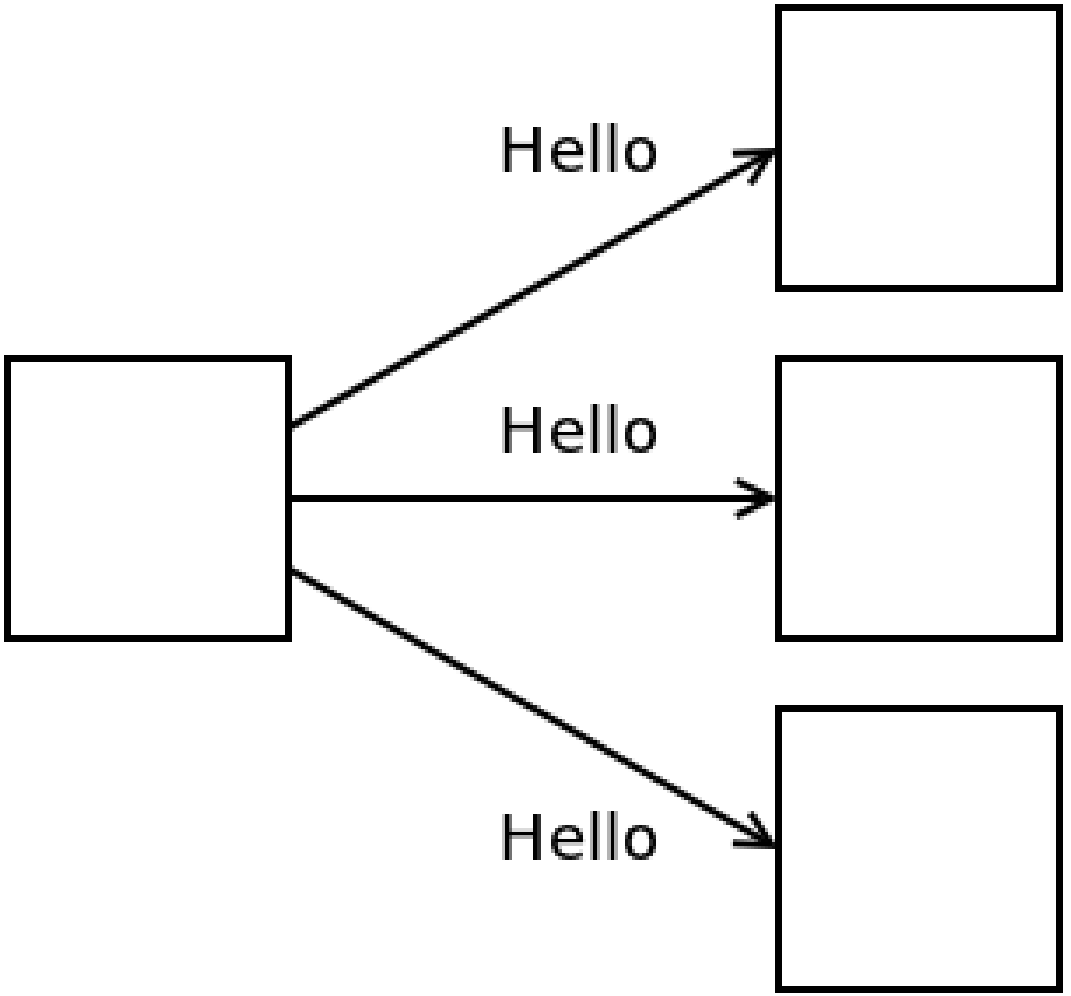
```
require 'rubygems'
require 'ffi-rzmq'

context = ZMQ::Context.new
socket = context.socket(ZMQ::REQ)
socket.connect('tcp://localhost:5559')

socket.send_string("Hello")
puts socket.recv_string ('')
```

```
require 'rubygems'
require 'ffi-rzmq'

context = ZMQ::Context.new
socket = context.socket(ZMQ::REP)
socket.bind("tcp://*:5559")

while true do
  request = socket.recv_string ('')
  socket.send_string("World")
end
```


REQ
REQ
REQ
REP

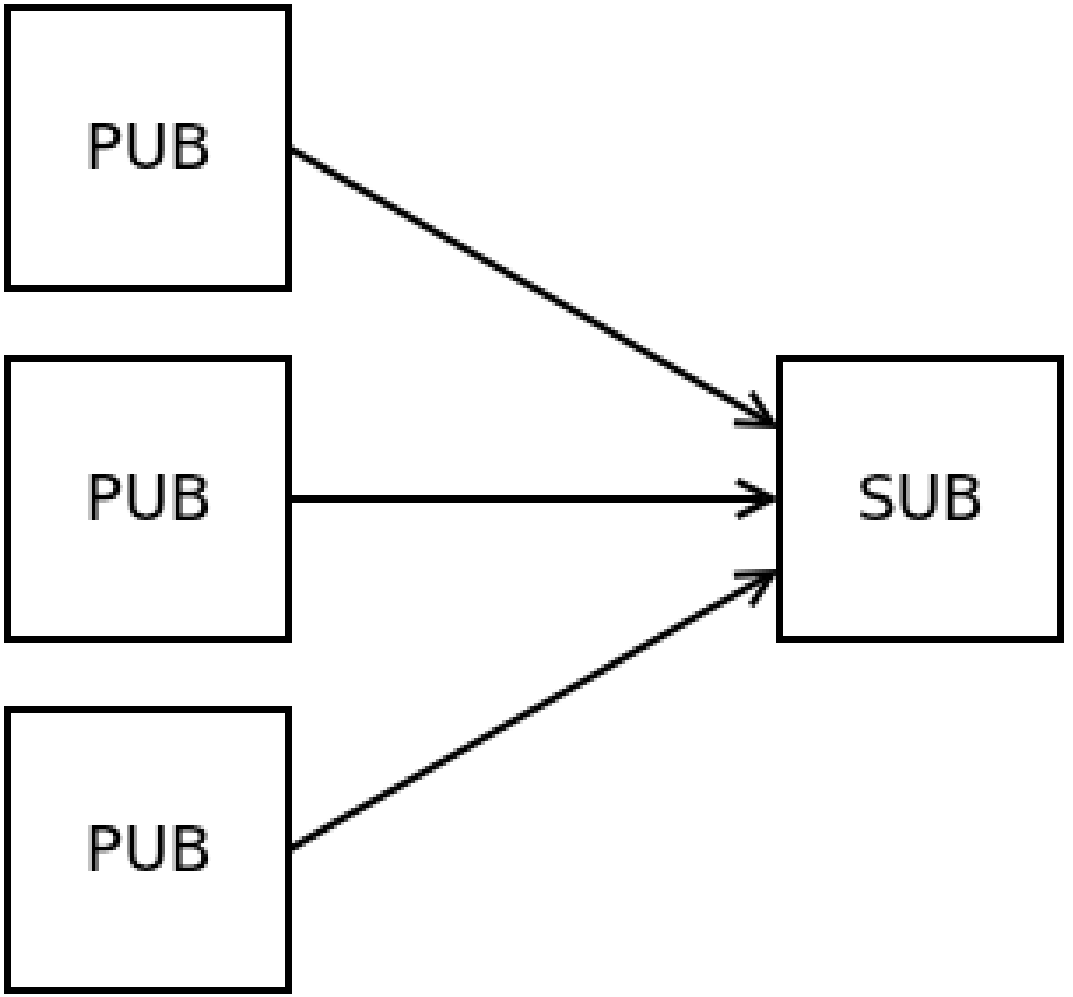

1, 4, 7, ...
REP
REQ
2, 5, 8, ...
REP
REP
3, 6, 9, ...

```
require 'rubygems'
require 'ffi-rzmq'

context = ZMQ::Context.new
socket = context.socket(ZMQ::REQ)
socket.connect('tcp://192.168.0.1:5559')
socket.connect('tcp://192.168.0.2:5559')
socket.connect('tcp://192.168.0.3:5559')

while true do
   socket.send_string("Hello")
   puts socket.recv_string ('')
end
```

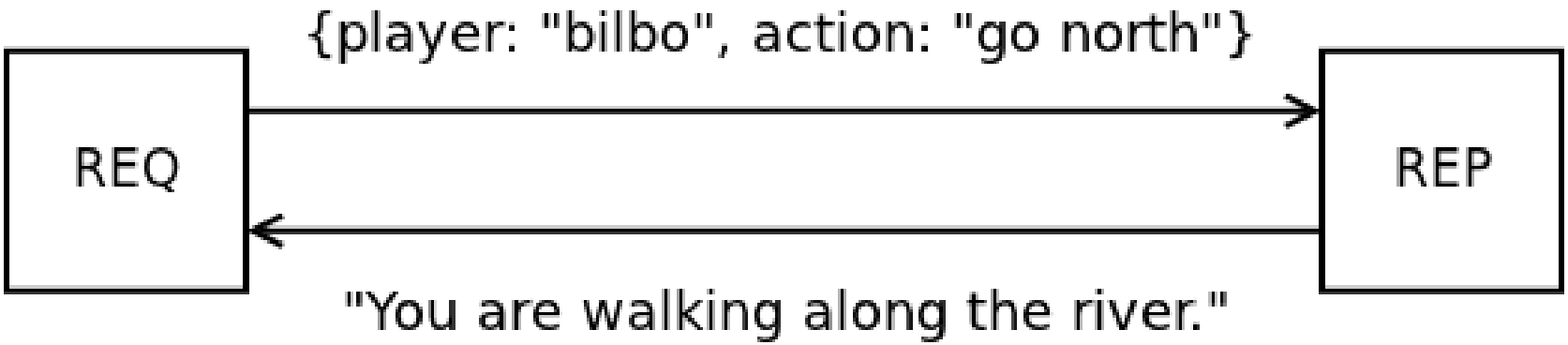

Hello
Hello
Hello

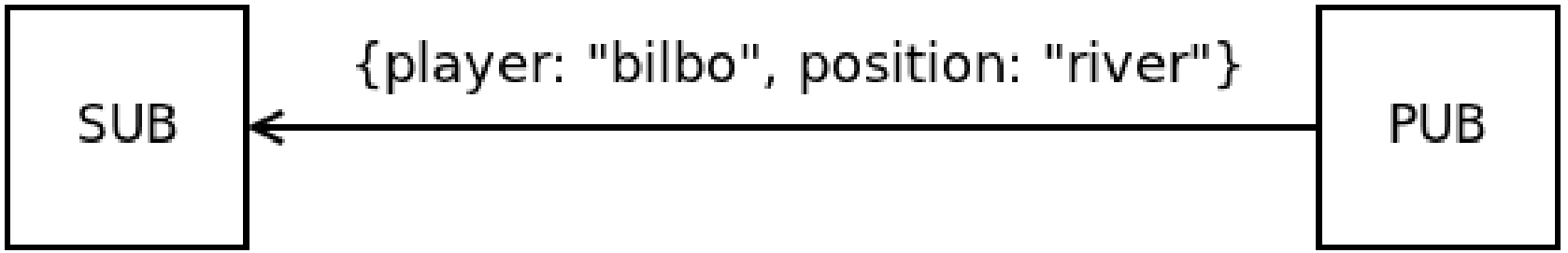
```
require 'rubygems'
require 'ffi-rzmq'

context = ZMQ::Context.new
socket = context.socket(ZMQ::PUB)
socket.bind("tcp://*:5559")

while true
   socket.send_string("Hello")
   sleep (1)
end
```

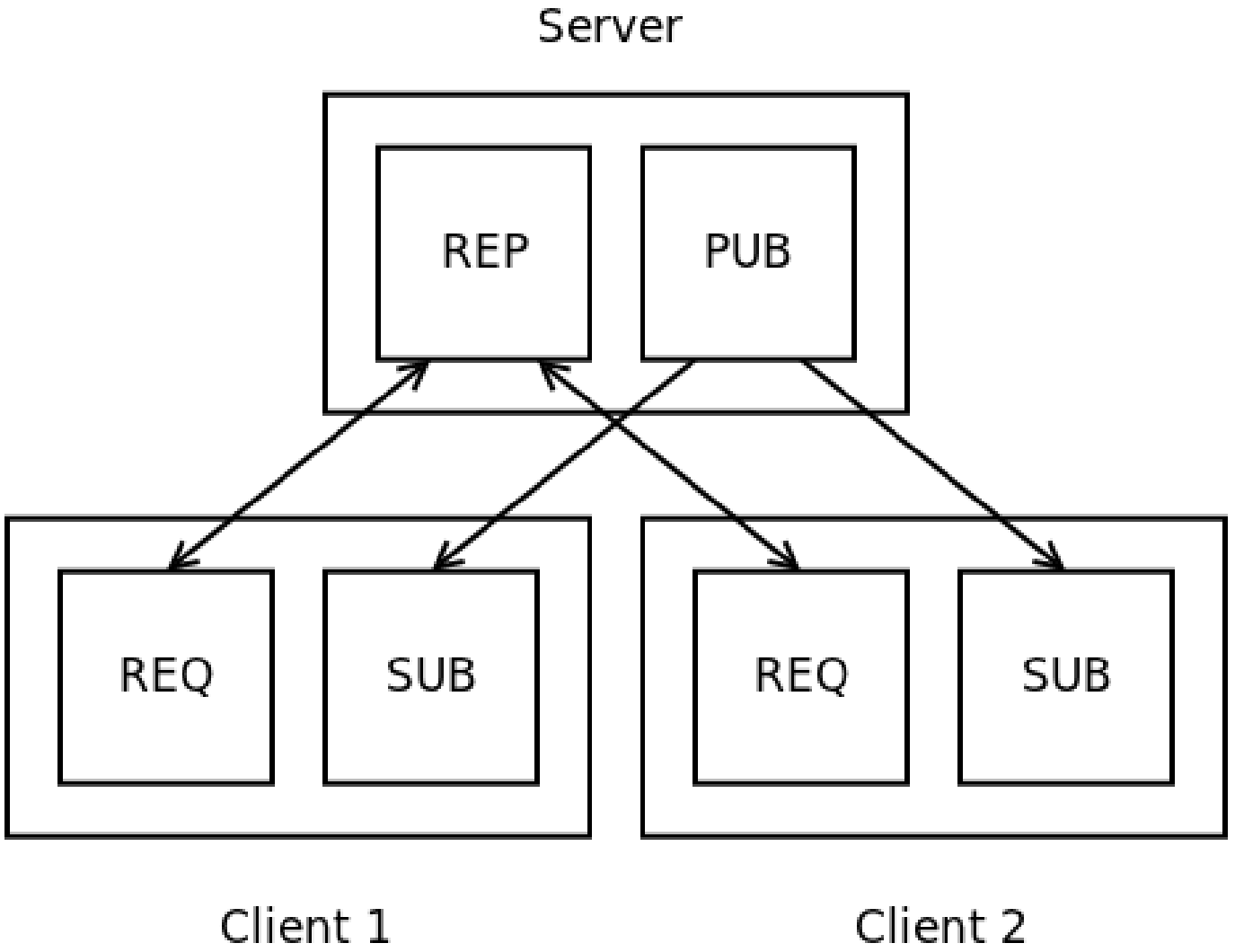
```
require 'rubygems'
require 'ffi-rzmq'

context = ZMQ::Context.new
socket = context.socket(ZMQ::SUB)
socket.connect("tcp://srv.example.org:5559")

socket.setsockopt(ZMQ::SUBSCRIBE, "")

while true do
   puts socket.recv_string ('')
end
```

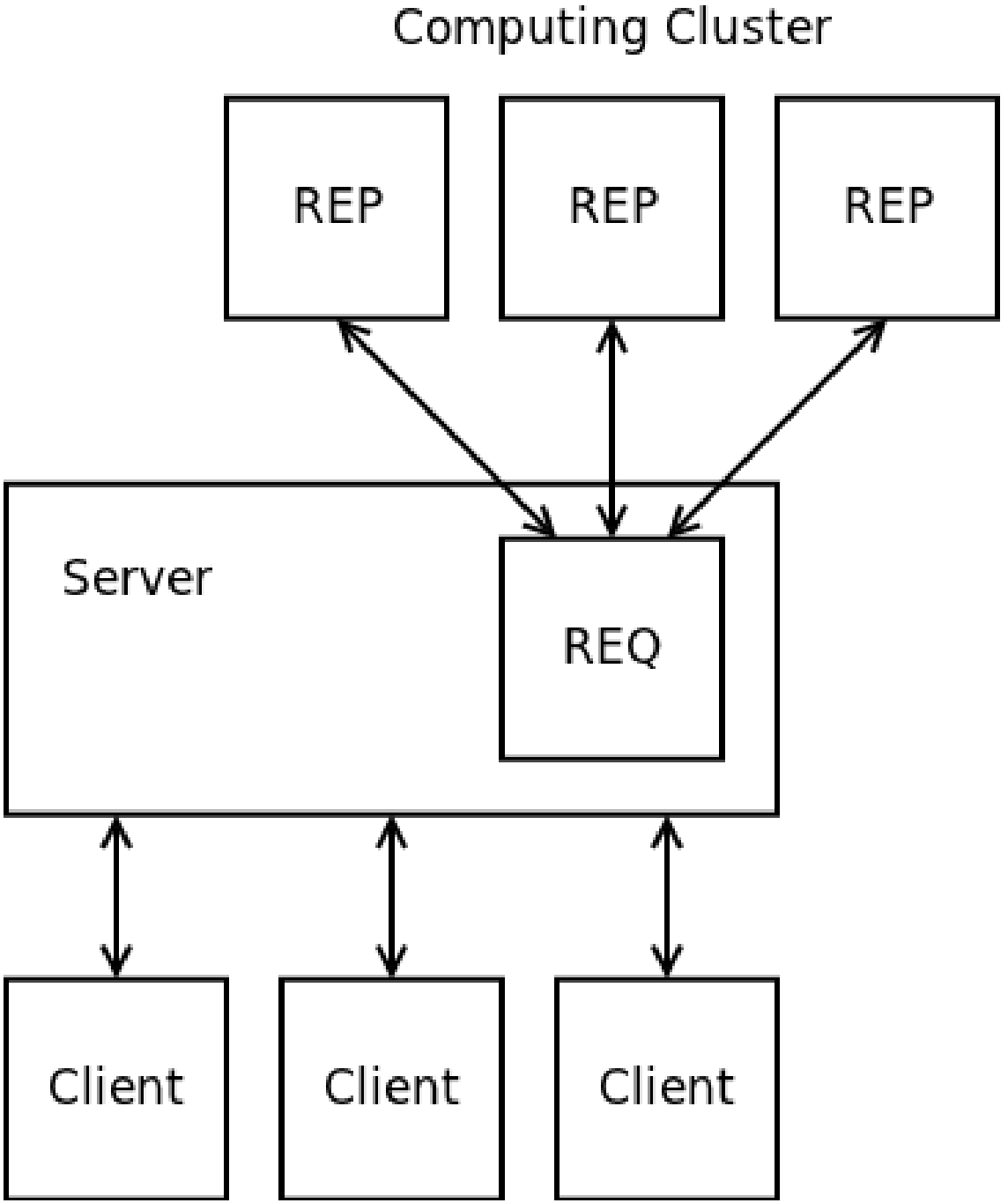

PUB
PUB
PUB
SUB

Example
Multiplayer Game


{player: "bilbo", action: "go north"}
REQ
REP
"You are walking along the river."

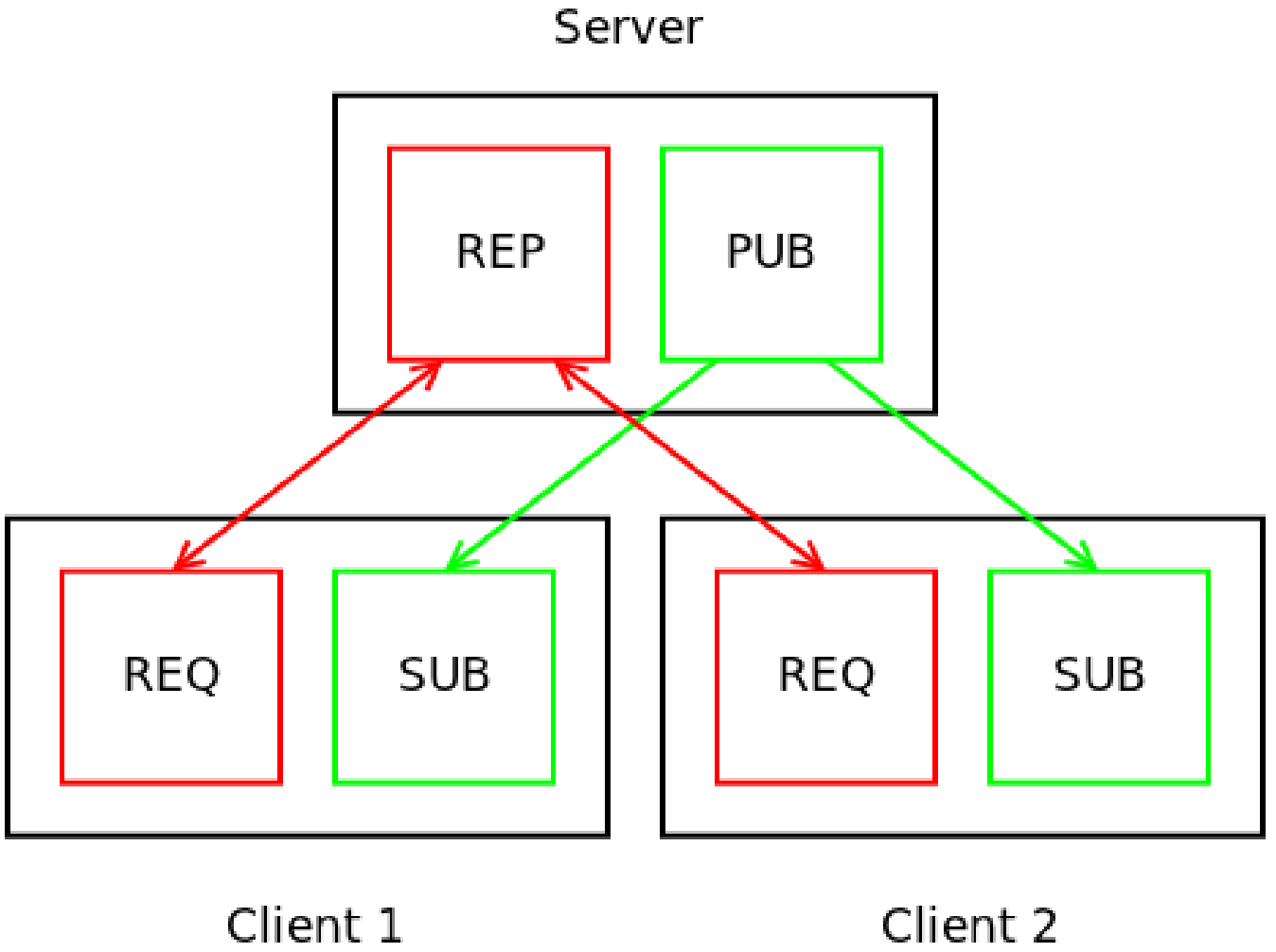

SUB
{player: "bilbo", position: "river"}
PUB


Server
REP
PUB
REQ
SUB
REQ
SUB
Client 1
Client 2


Computing Cluster
REP
REP
REP
Server
REQ
Client
Client
Client

The Administrative Aspect

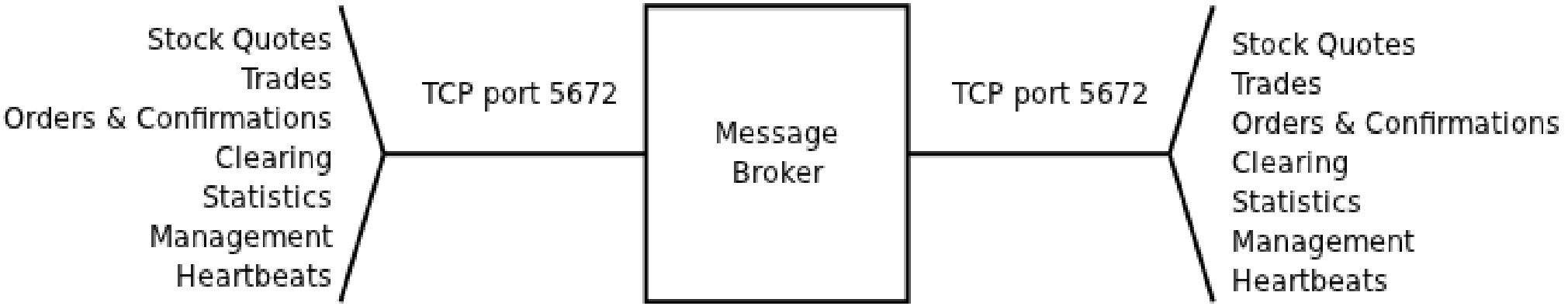
Message Feeds and Topologies



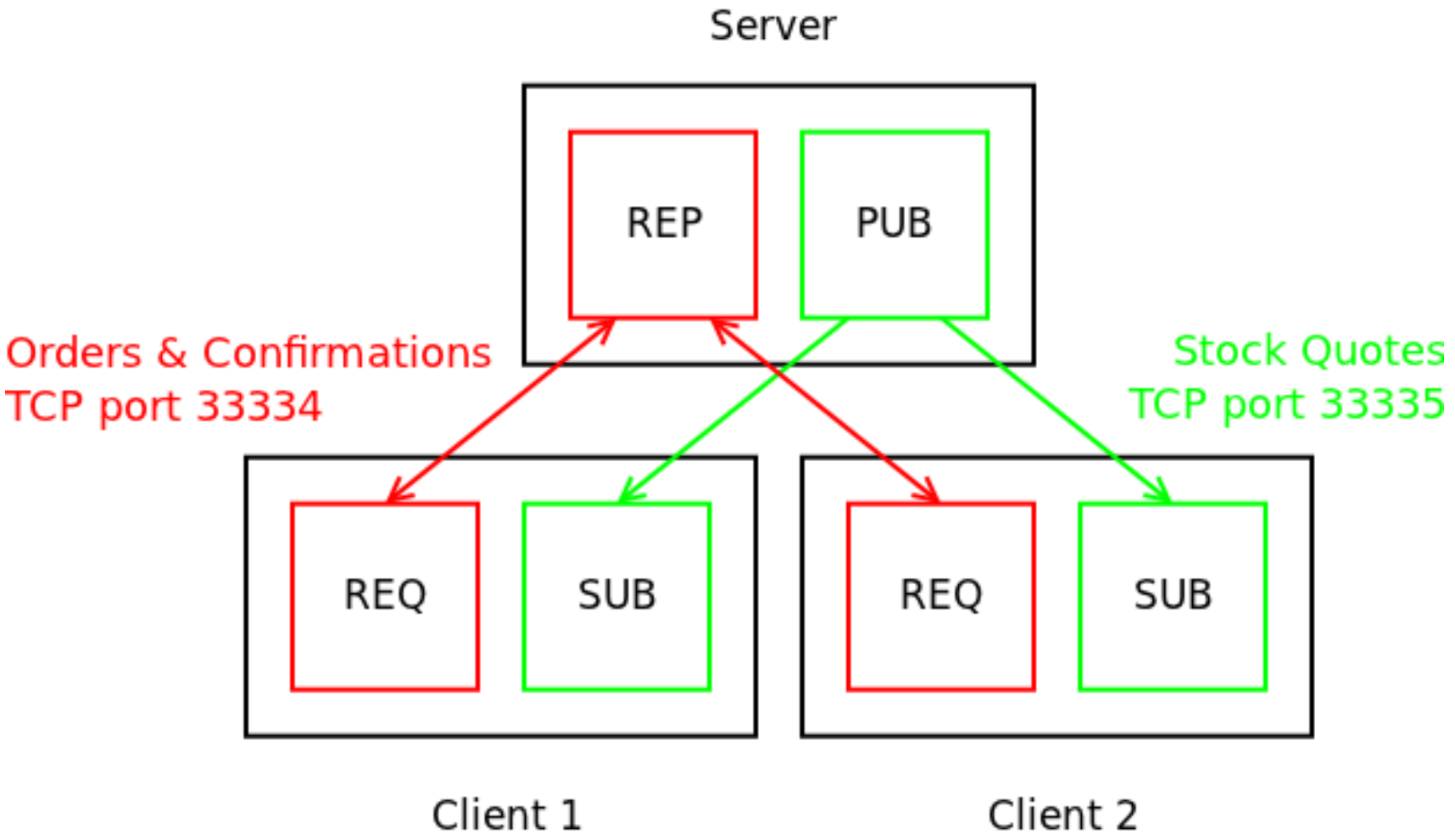
Example: Stock Exchange

- Stock Quotes
- Trades
- Orders & Order Confirmations
- Clearing
- Statistics
- Management
- Heartbeats

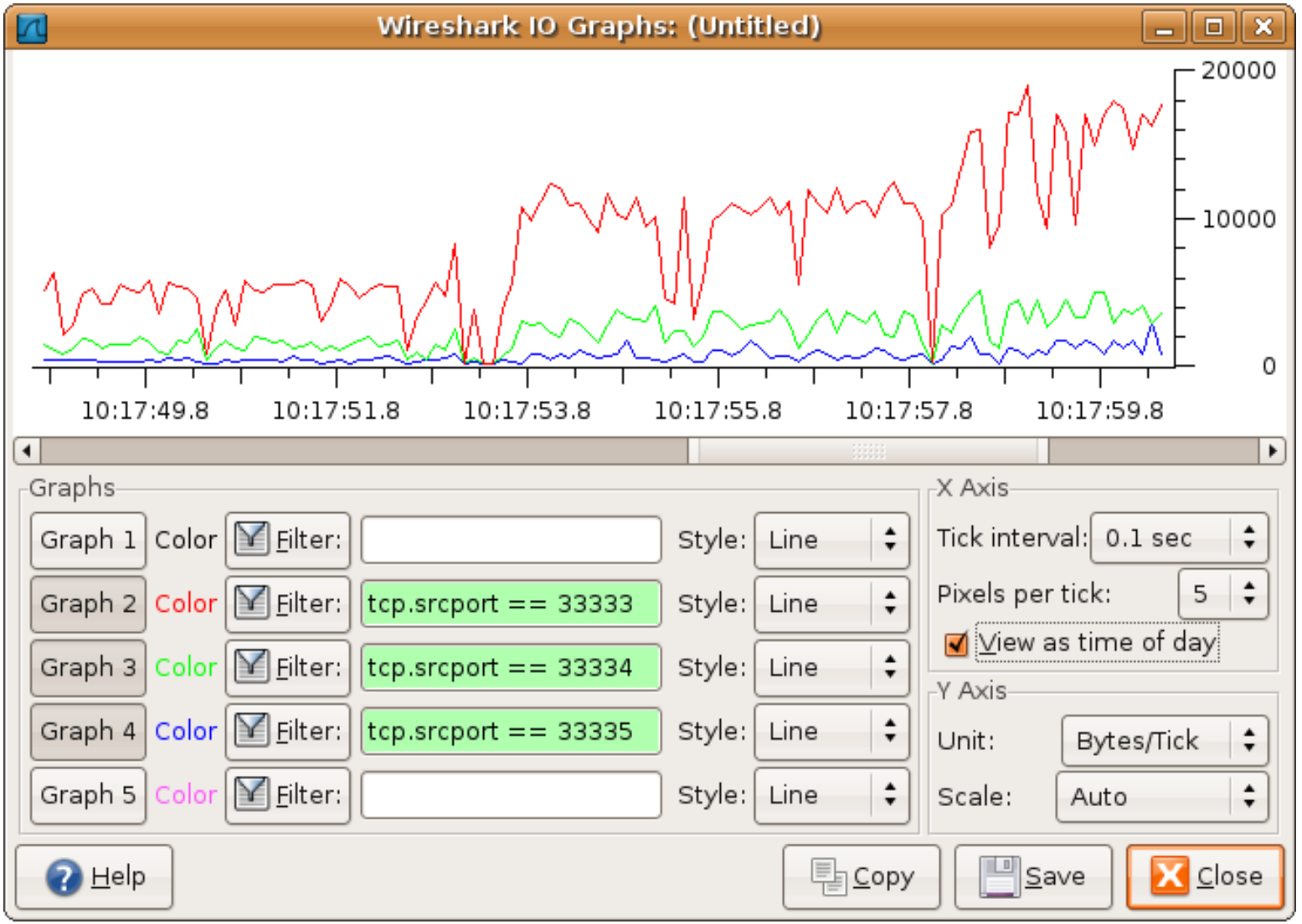
Centralised Messaging



Distributed Messaging



Monitoring

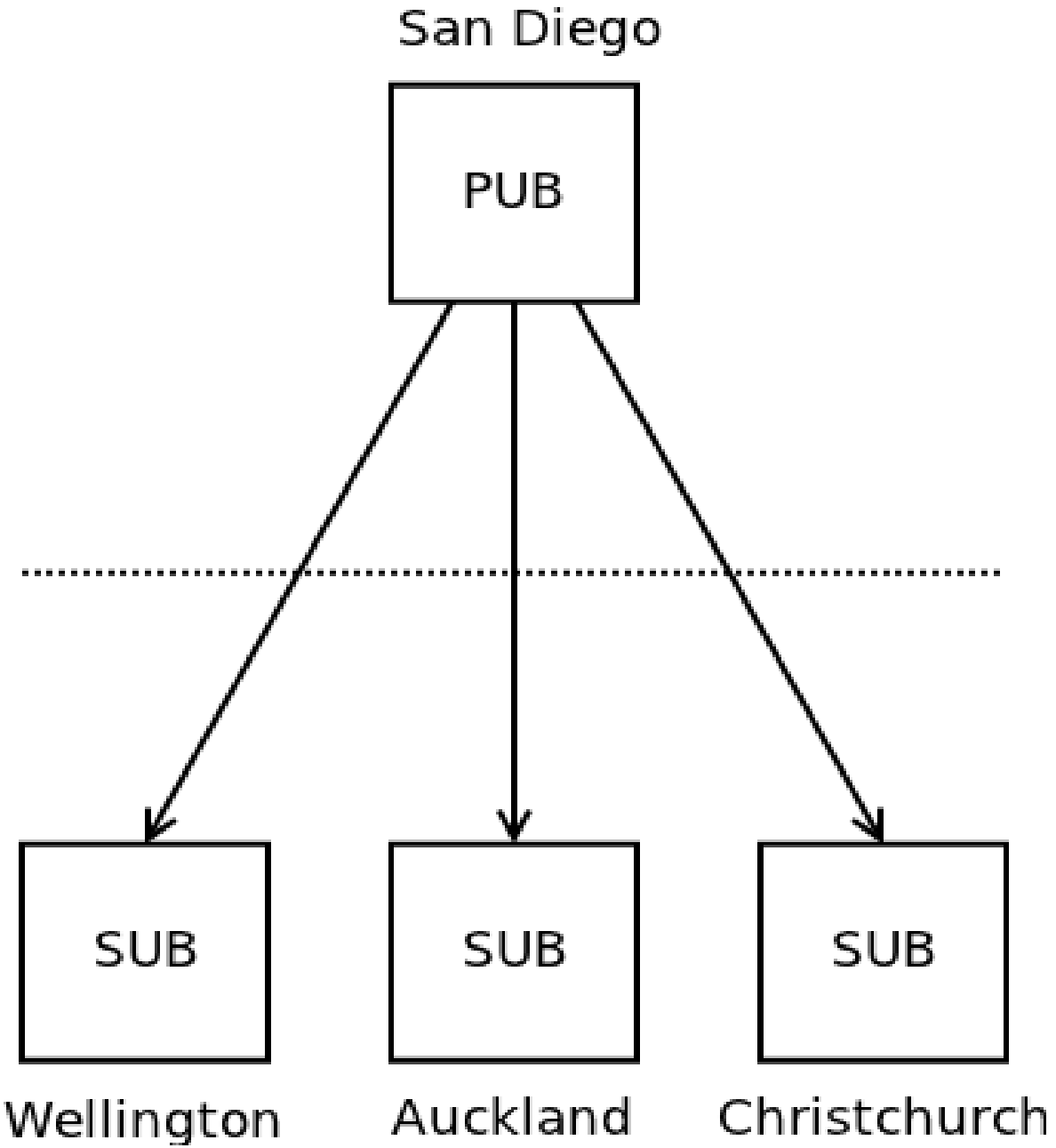

Wireshark IO Graphs: (Untitled)
20000
10000
0
10:17:49.8
10:17:51.8
10:17:53.8
10:17:55.8
10:17:57.8
10:17:59.8
Graphs
Graph 1 Color Filter: Style: Line
Graph 2 Color Filter: tcp.srcport == 33333 Style: Line
Graph 3 Color Filter: tcp.srcport == 33334 Style: Line
Graph 4 Color Filter: tcp.srcport == 33335 Style: Line
Graph 5 Color Filter: Style: Line
X Axis
Tick interval: 0.1 sec
Pixels per tick: 5
View as time of day
Y Axis
Unit: Bytes/Tick
Scale: Auto
Help
Copy
Save
Close

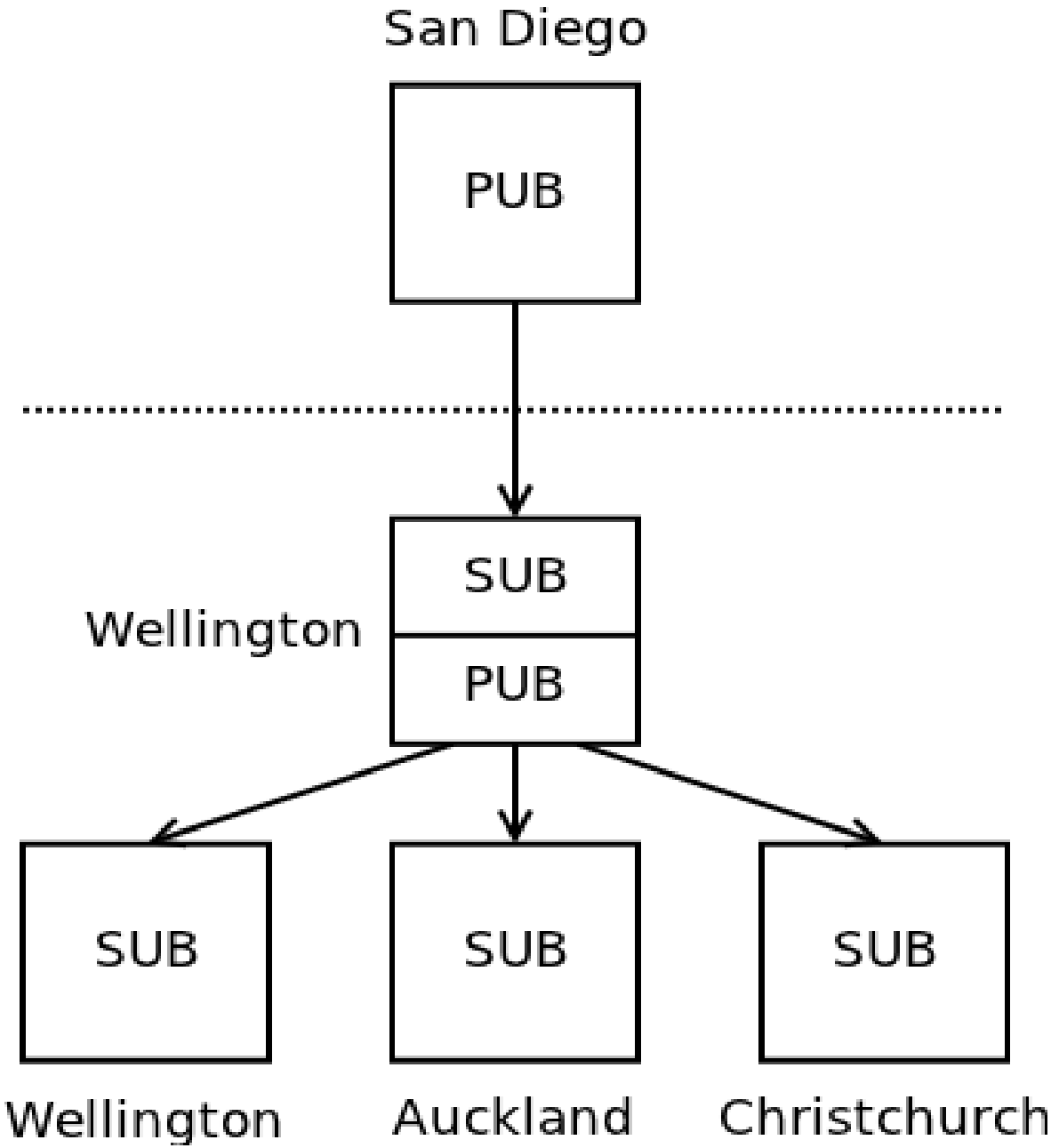
Traffic Shaping

TCP port 33333, i.e. stock quotes

```
access-list 100 permit tcp any any eq 33333
class-map match-any port33333
  match access-group 100
policy-map port33333
  class port33333
   bandwidth 5120
interface Ethernet0
 service-policy output port33333
```

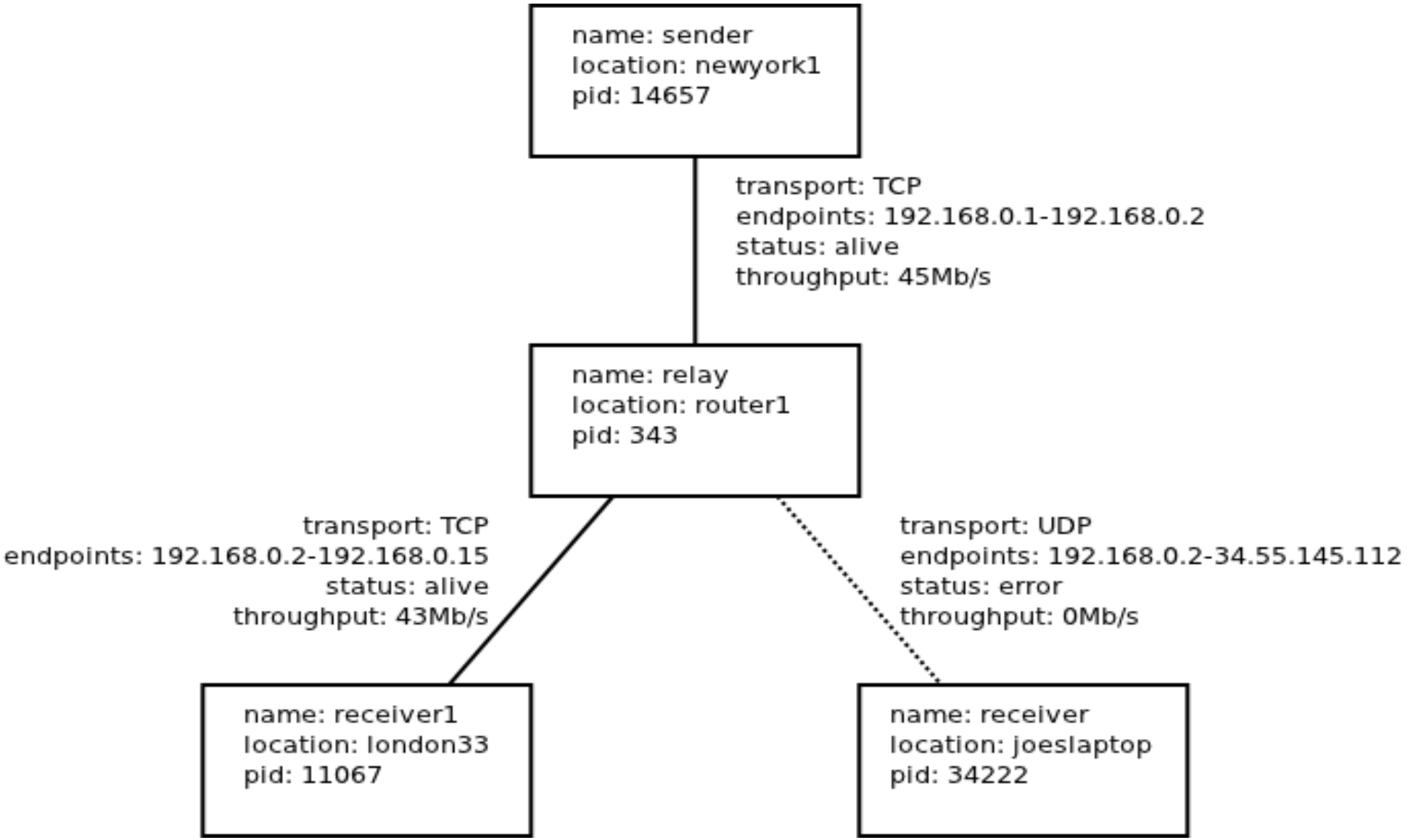
5120 kb/sec


San Diego
PUB
SUB
SUB
SUB
Wellington
Auckland
Christchurch

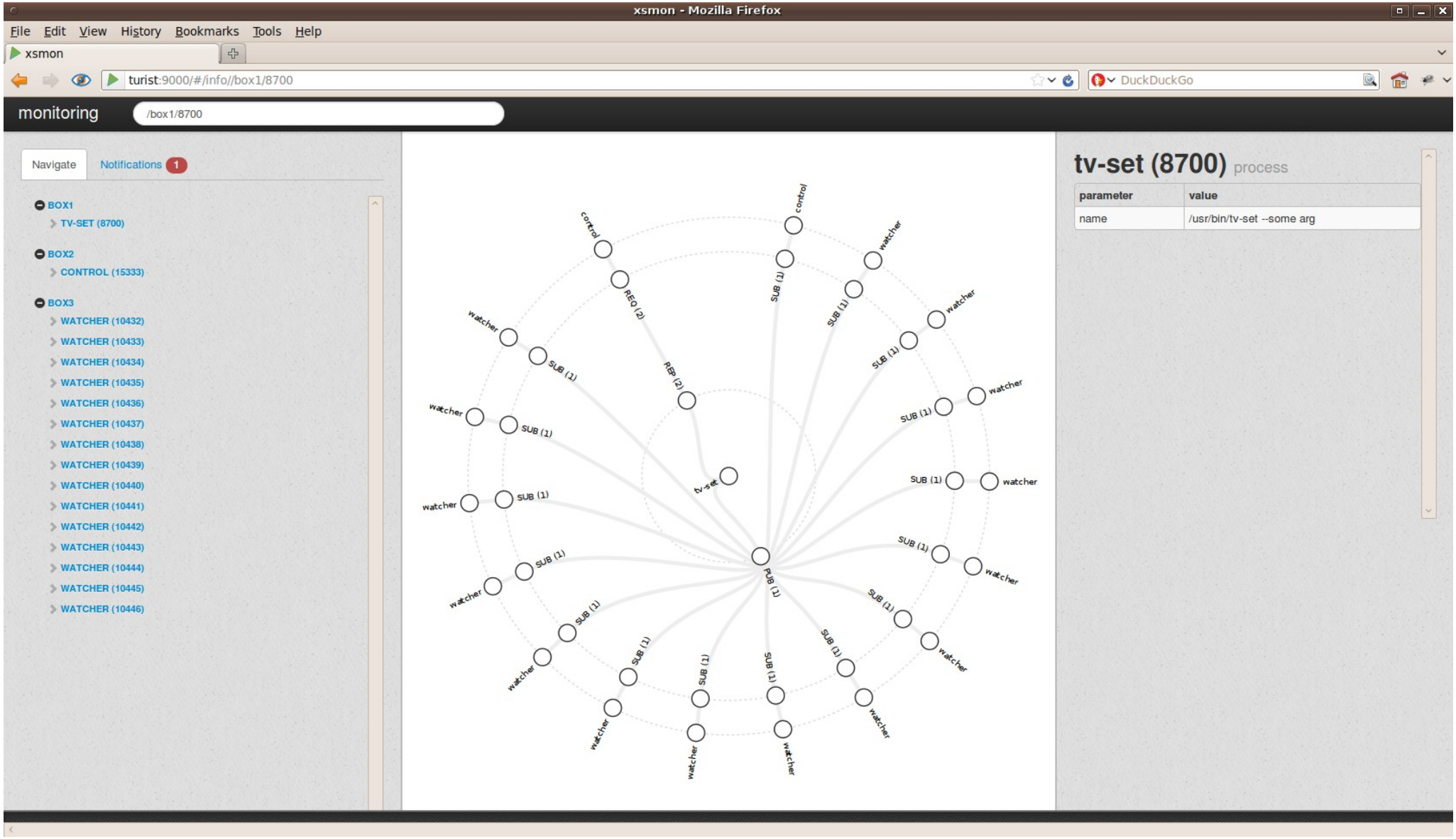

San Diego
PUB
SUB
Wellington
PUB
SUB
SUB
SUB
Wellington
Auckland
Christchurch

What's next?

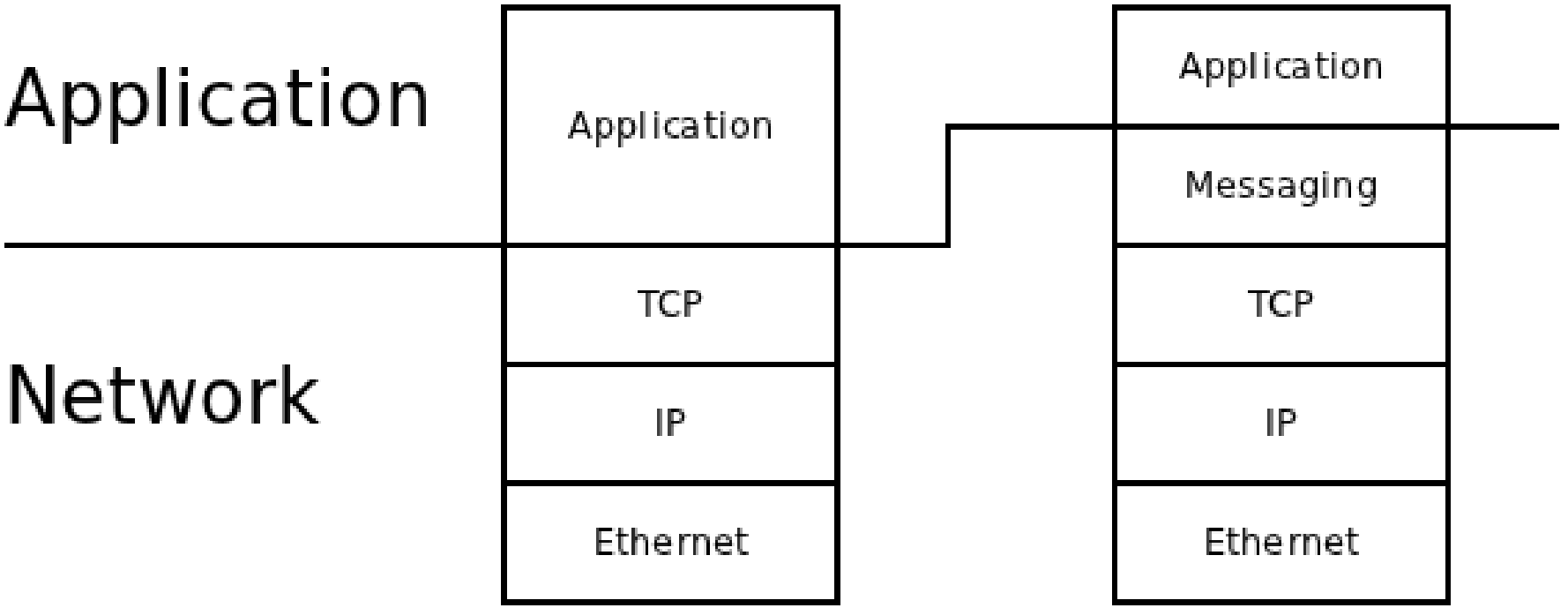
Monitoring?


name: sender
location: newyork1
pid: 14657
transport: TCP
endpoints: 192.168.0.1-192.168.0.2
status: alive
throughput: 45Mb/s
name: relay
location: router1
pid: 343
transport: TCP
endpoints: 192.168.0.2-192.168.0.15
status: alive
throughput: 43Mb/s
transport: UDP
endpoints: 192.168.0.2-34.55.145.112
status: error
throughput: 0Mb/s
name: receiver1
location: london33
pid: 11067
name: receiver
location: joeslaptop
pid: 34222

Monitoring?


xsmon - Mozilla Firefox
File Edit View History Bookmarks Tools Help
xsmon
turist:9000/#/info//box1/8700
DuckDuckGo
monitoring
/box1/8700
Navigate
Notifications 1
BOX1
TV-SET (8700)
BOX2
CONTROL (15333)
BOX3
WATCHER (10432)
WATCHER (10433)
WATCHER (10434)
WATCHER (10435)
WATCHER (10436)
WATCHER (10437)
WATCHER (10438)
WATCHER (10439)
WATCHER (10440)
WATCHER (10441)
WATCHER (10442)
WATCHER (10443)
WATCHER (10444)
WATCHER (10445)
WATCHER (10446)
tv-set (8700) process
parameter
value
name
/usr/bin/tv-set --some arg
control
watcher
SUB (1)
REQ (2)
REP (2)
tv-set
PUB (1)

Hardware?


Application
Network
Application
TCP
IP
Ethernet
Application
Messaging
TCP
IP
Ethernet

Distibuted Debugger?

Questions?

Martin Sústrik
www.250bpm.com
sustrik@250bpm.com