

# Request-Reply

# Antes de mais nada

- Não vamos usar sockets
- Mas vamos usar sockets:
  - Nível mais baixo do que vimos antes
  - Mas não tão baixo quanto em S.O.

# Advanced Message Queuing Protocol (AMQP)

- Projeto inicial em 2003
- Protocolo para troca de mensagens
- Possui algumas implementações: Apache Qpid, RabbitMQ, Apache ActiveMQ, IBM MQ
- Documentação disponível em <https://zguide.zeromq.org/>

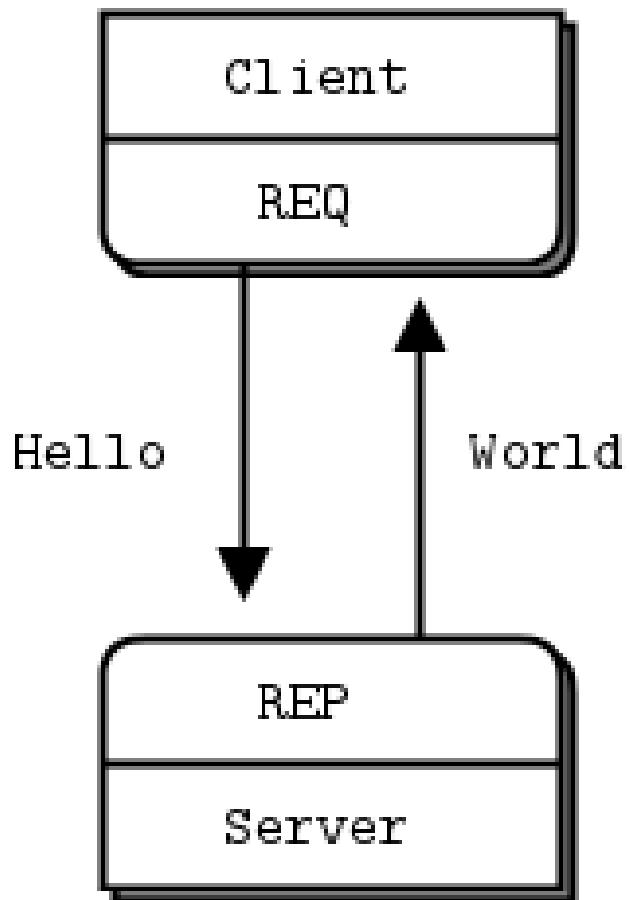
# ZeroMQ

- Mais simples que o AMQP
- Permite a construção de diversos padrões (esta e próximas aulas)
- Podemos escolher entre 28 linguagens
- Curiosidade: usado pelo CERN para a troca de informações entre os aceleradores de partículas
- Vamos fazer (quase tudo) na mão desta vez

# **Padrões de troca de mensagem**

- Request-reply: cliente envia mensagem (request) e servidor responde (reply)
- Publish-subscribe: servidor publica (publish) e clientes se inscrevem (subscribe) para receber as mensagens
- Pipeline: cliente envia mensagem, workers trabalham em paralelo na mensagem, um cliente recebe o resultado (dividir e conquistar)

# Request-reply (com ØMQ)



- conexão via socket entre os processos
- servidor abre uma conexão do tipo `zmq.REP`
- cliente se conecta usando `zmq.REQ`
- cliente e servidor trocam mensagem em bytes
- **cliente inicia enviando mensagem para o servidor**

# Para testar

- Verificar os arquivos na pasta `src/reqrep`
- O `Dockerfile` é o mesmo que será usado nas próximas aulas. Só estará disponível para facilitar a criação das imagens
- Para executar: `docker compose up`

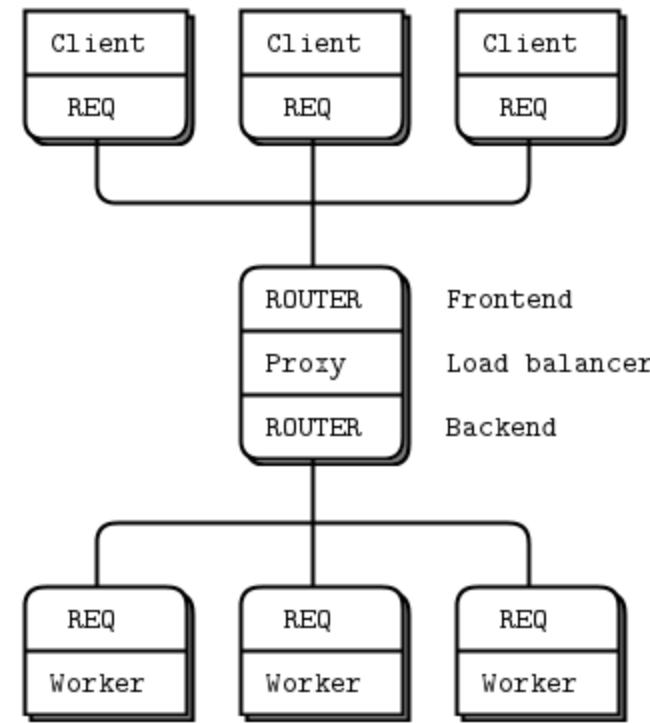
# Alguns testes

- E se inverter a ordem? (trocar o `depends_on` do `docker-compose.yml`)
- E ser o servidor cair e voltar?
- O que é o `b` antes da `string`?
- O que acontece se aumentarmos a quantidade de réplicas do cliente?
- O que acontece se aumentarmos a quantidade de réplicas do servidor?

# Outras combinações de conexão

- Request - Reply
- Dealer - Reply
- Request - Router
- Dealer - Router
- Dealer - Dealer
- Router - Router
- Dealer: tipo request, mas assíncrono
- Router: tipo reply, mas assíncrono

Padrão de平衡amento de carga



# Usando broker com docker

- Verificar os arquivos na pasta `src/broker`
- O `Dockerfile` e a imagem criada são as mesmas do exemplo anterior
- Para executar: `docker compose up`

# Mais coisas para testar

- Porque os dois dependem do broker?
- O que acontece se aumentarmos a quantidade de réplicas do cliente?
- O que acontece se aumentarmos a quantidade de réplicas do servidor?