Workshop

Practical Project

Comparing performance of IO frameworks

Institut d'informatique
Université de Neuchâtel

redhat.

Aliya Ibragimova
Teodor Macicas
Jean-Frederic Clere

Agenda

- Introduction
- I/O frameworks
- Testing tool
  - problems
  - architecture
  - test scenarios
- Data processing
- Results
- Conclusion

Introduction

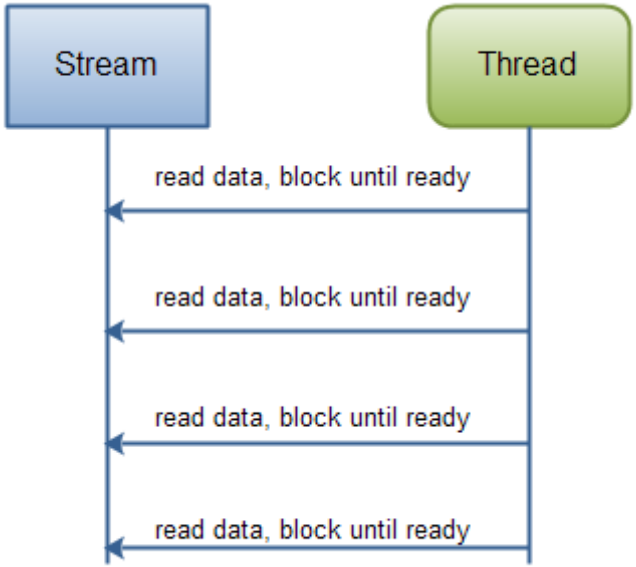
Input/Output (I/O) frameworks:

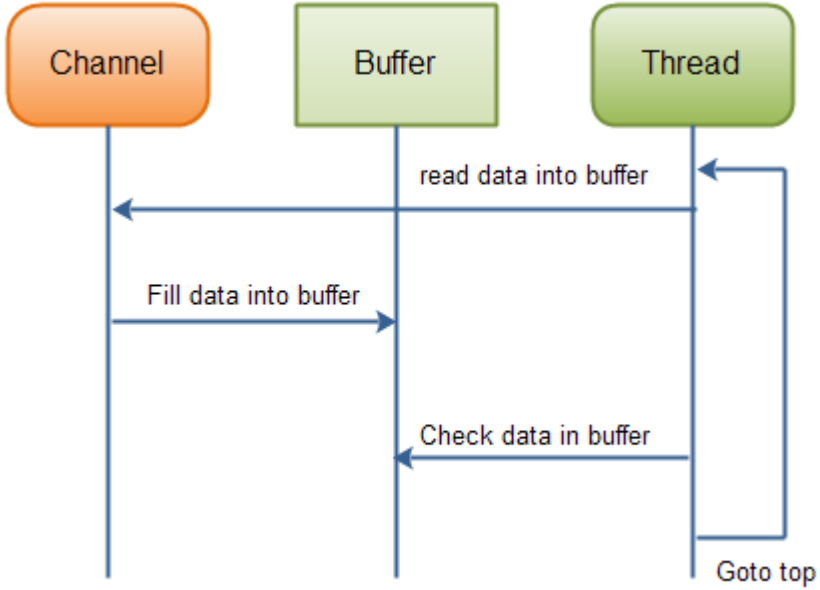
- abstractions for low-level operations
- features for intensive I/O operations

NIO (New I/O in Java 1.4)

I/O vs NIO

I/O	NIO
Stream oriented	Buffer oriented
Blocking I/O	Non-blocking I/O
	Selectors

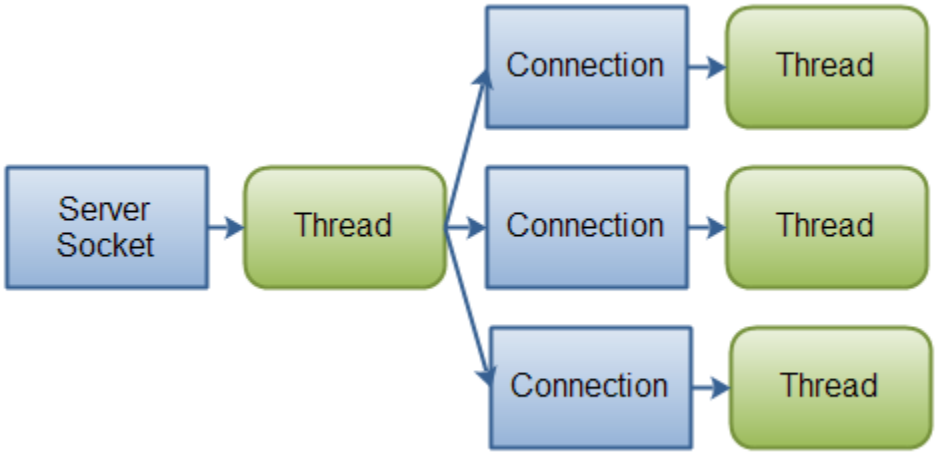




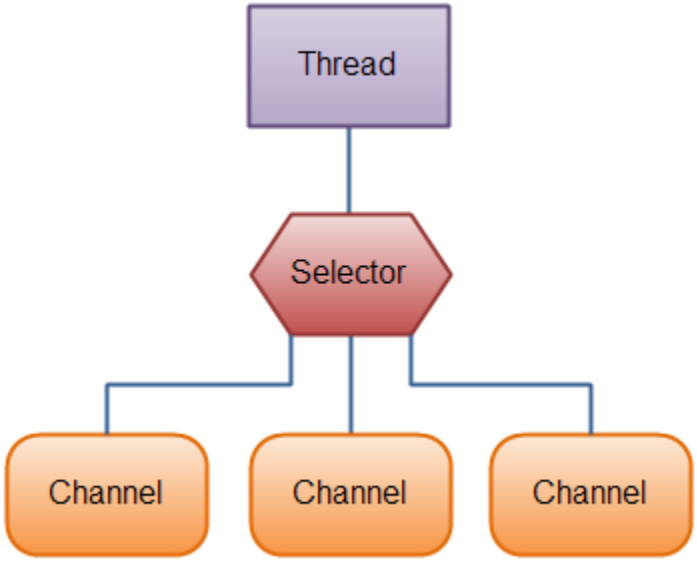
Source: http://tutorials.jenkov.com/java-nio/

I/O vs NIO

I/O: One thread per each connection approach



NIO: A thread uses a selector for handling 3 Channels



Limitations: lack of asynchronous API, not really easy to be used

Source: http://tutorials.jenkov.com/java-nio/

NIO2 - enhancement of NIO

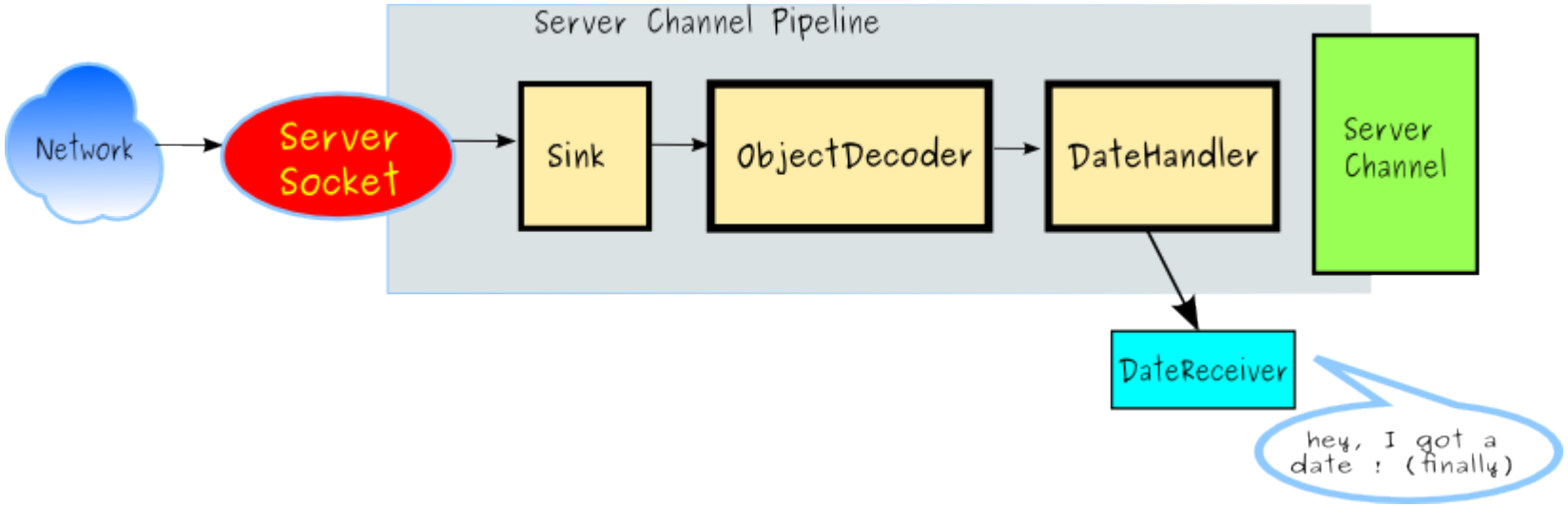
- Asynchronous channels are introduced:
- Asynchronous calls approaches:
  - Future style
  - Callback style
- New File-System API

XNIO3 - NIO improved

- Thread management
  - I/O threads
  - Worker threads
- More advanced framework
  - "not for beginners" (source: https://docs.jboss.org/author/display/XNIO/About+XNIO)
- Asynchronous (using callbacks)
- Blocking and non-blocking methods can be mixed even for the same Channel
- Zero-copy transfers

Netty - built on top NIO

- More abstract, so ease of use
- Every method is asynchronous
- Uses events and callbacks



Source: http://seeallhearall.blogspot.ch/2012/05/netty-tutorial-part-1-introduction-to.html

Project goals

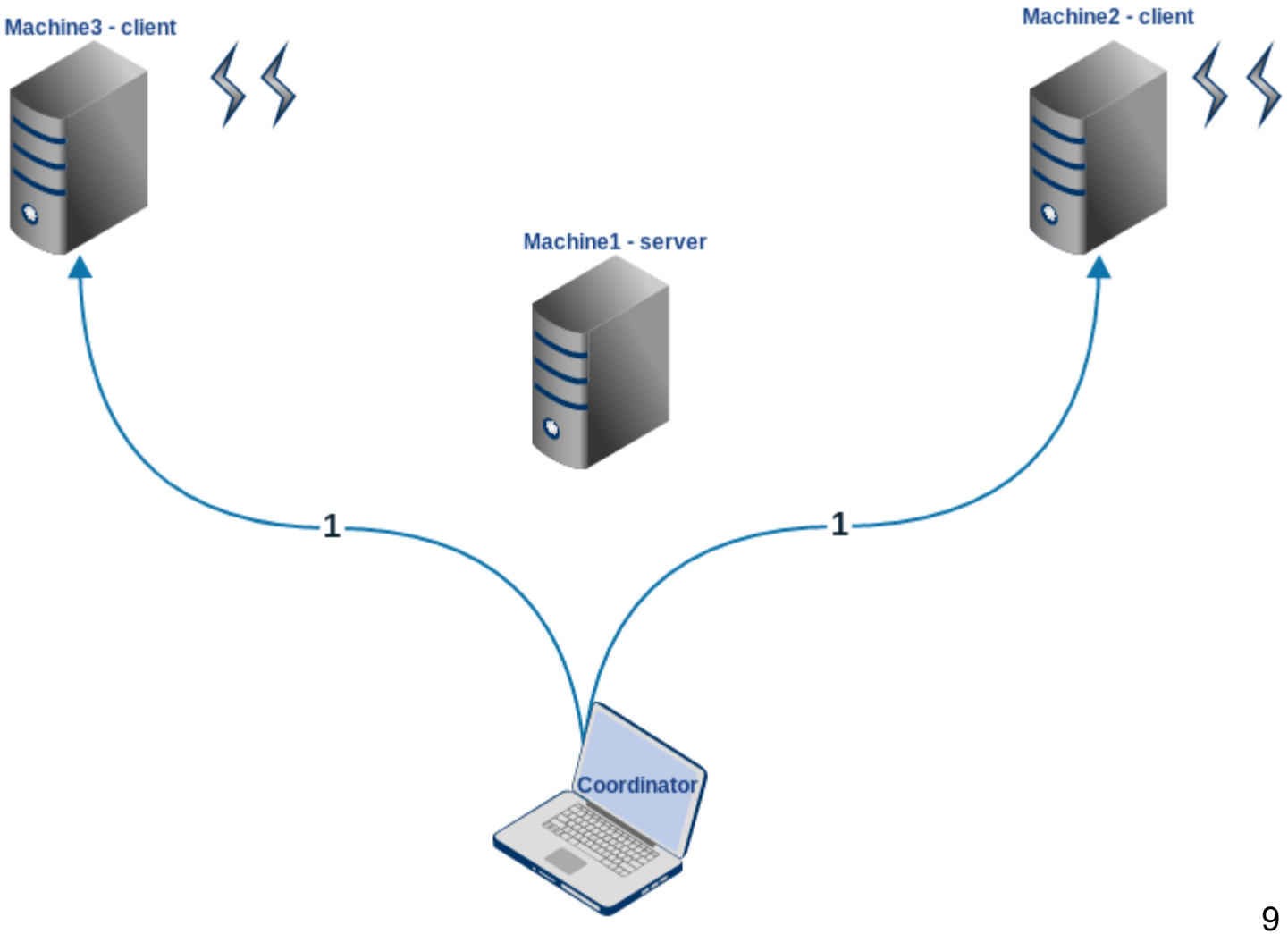
- Automate tests done in previous work and gathering statistics
  - Response time
  - CPU usage
  - Memory usage
- Add Netty framework to the tests
- Run tests and evaluate results

Testing tool - problem

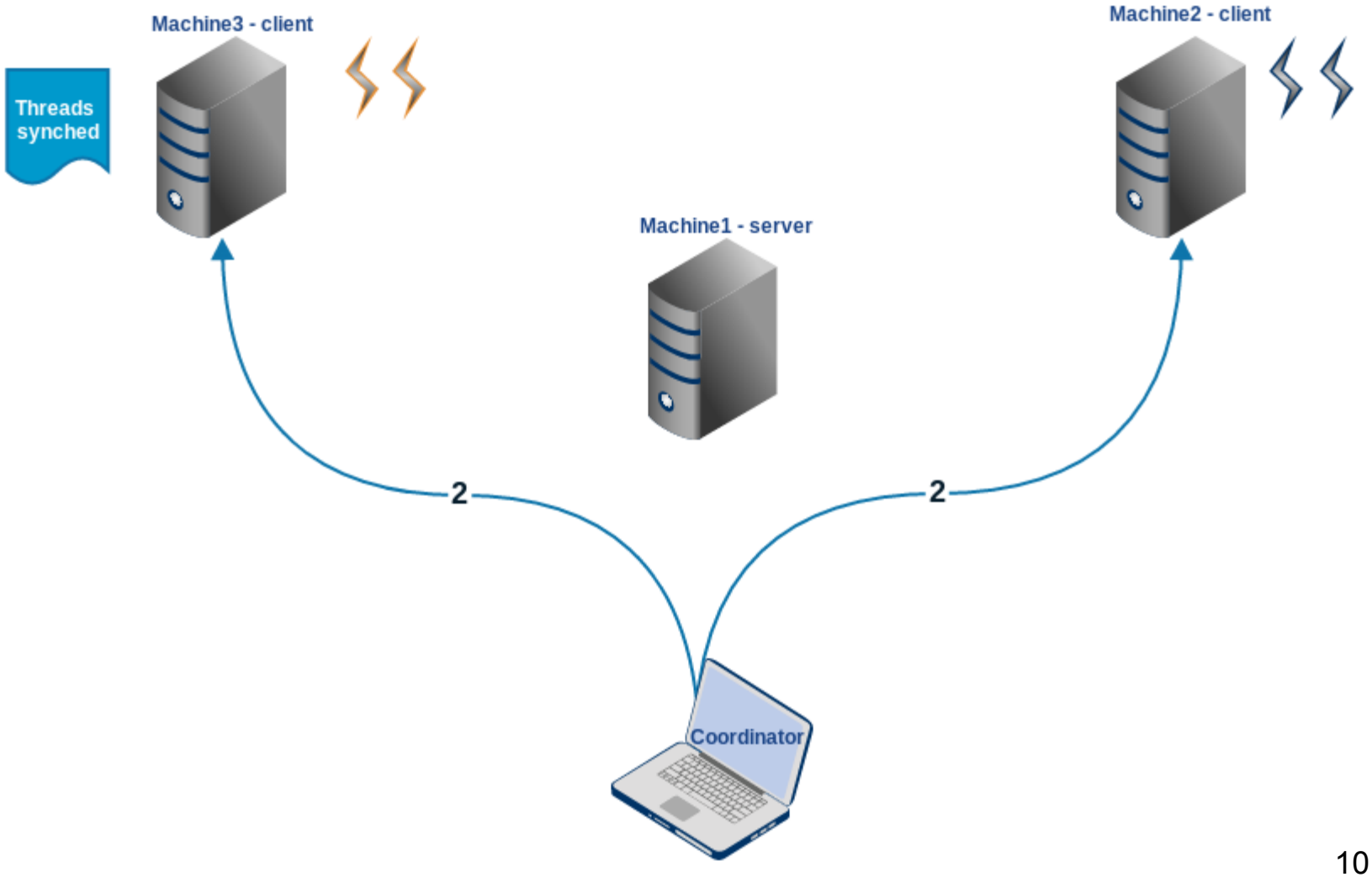
How to synchronize remote clients to ensure a bound of number of requests at server side?

1. Synchronize the threads of a client
   - solution: wait for a condition; main thread notifies them
2. Synchronize all the remote clients
   - solution: a 2PC-like approach
   - remote messages via local files

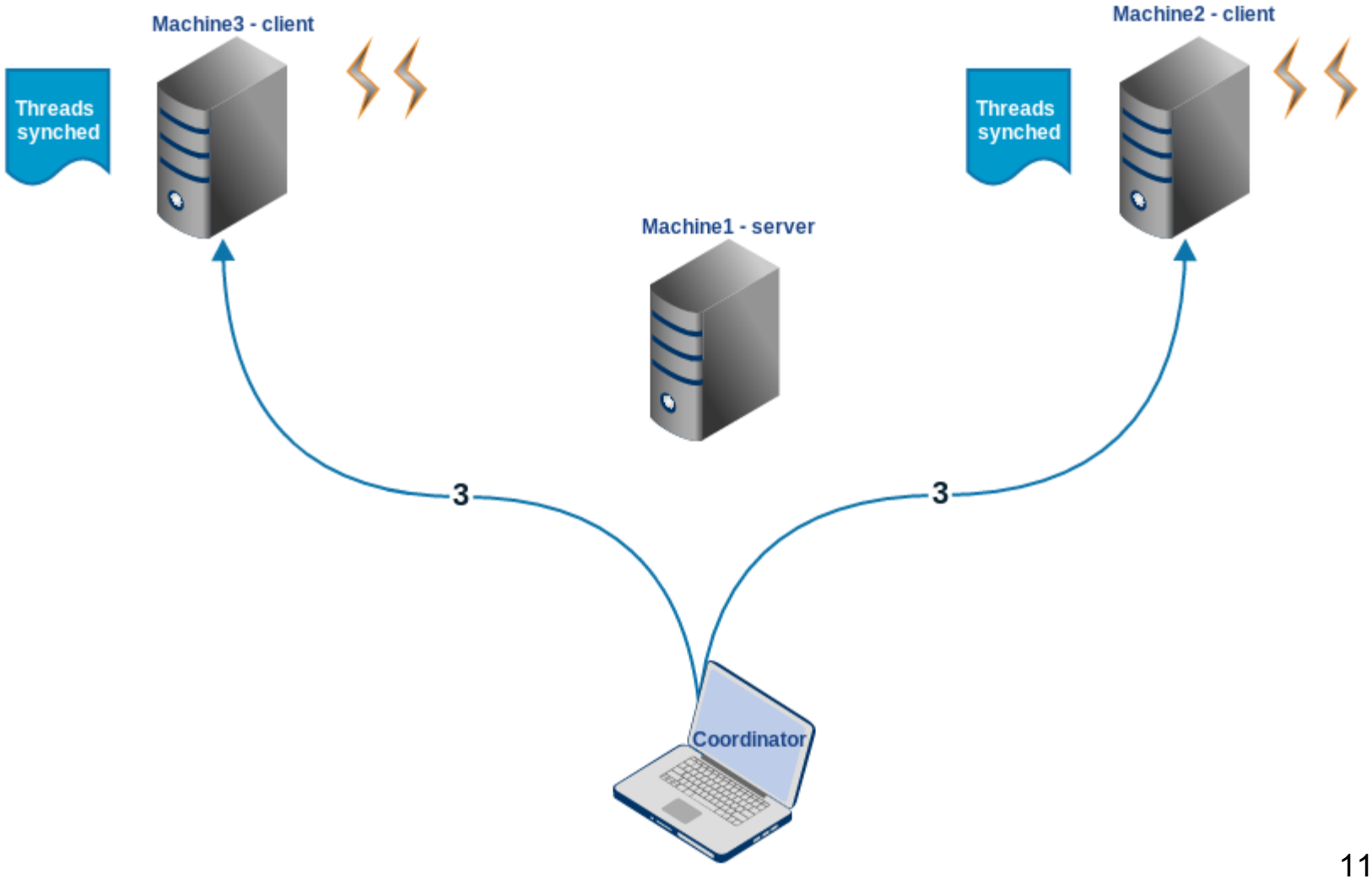
Testing tool - start threads (1)



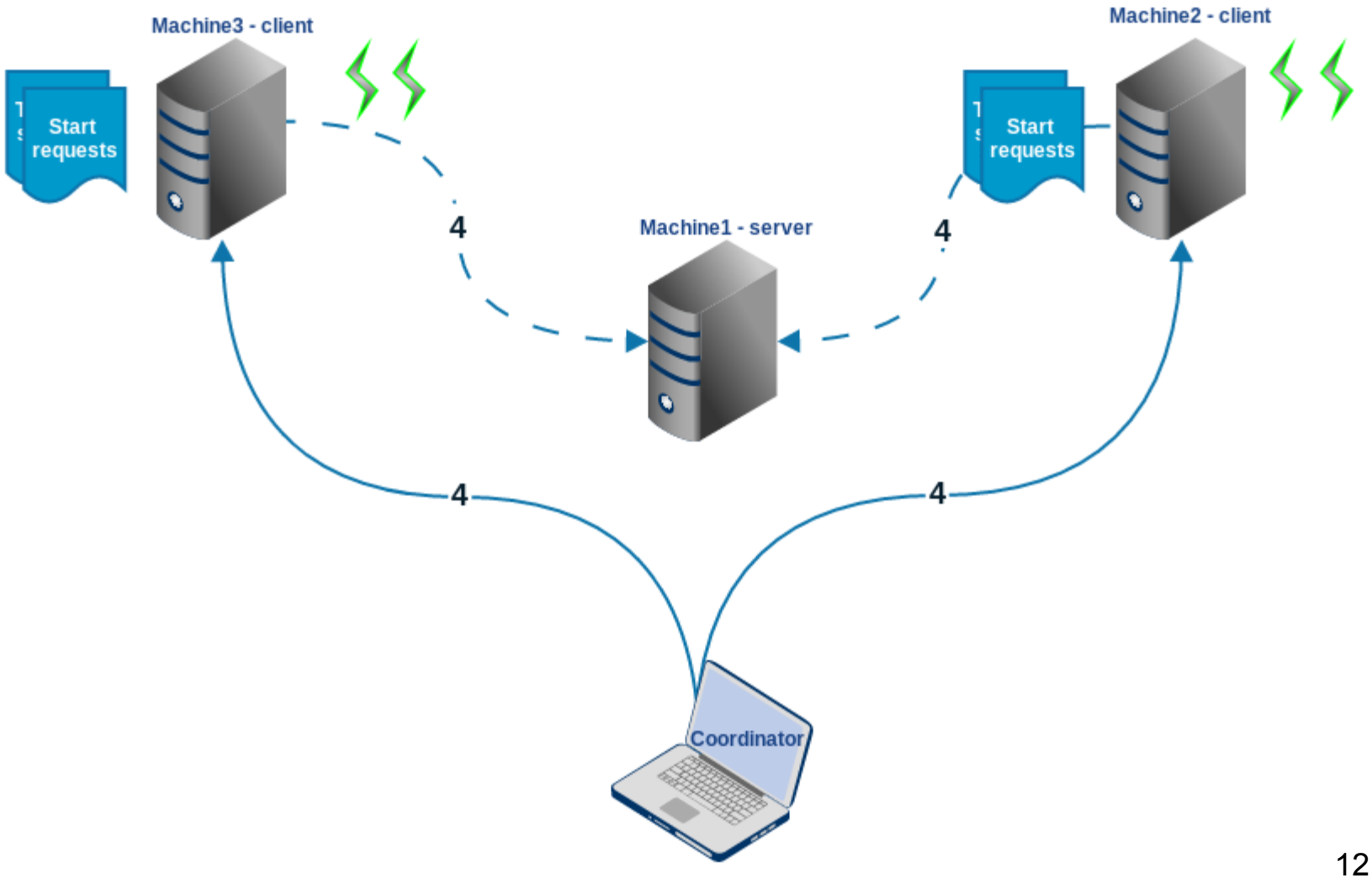
Testing tool - threads synched (2)



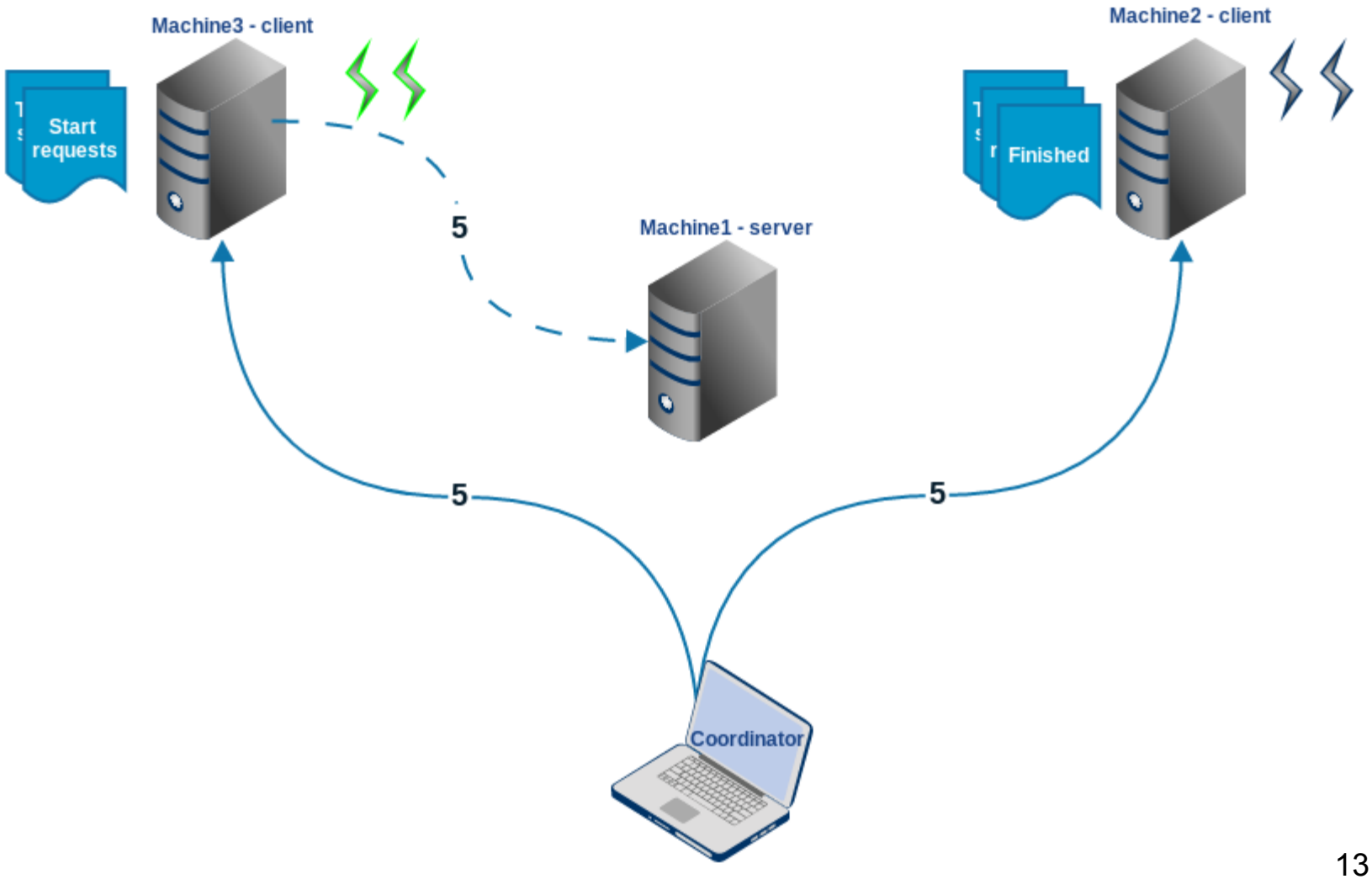
Testing tool - threads synched (2)



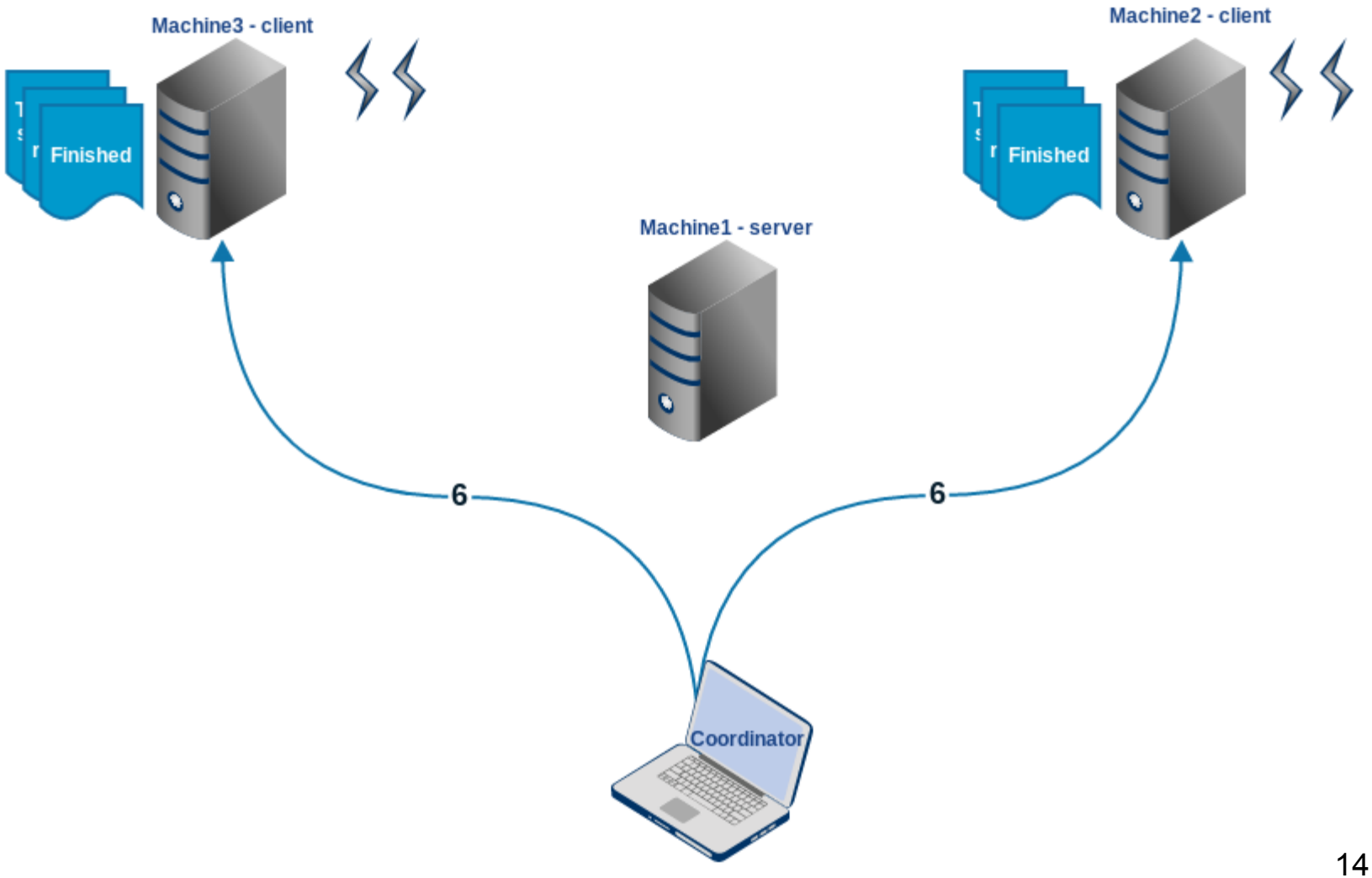
Testing tool - start sending requests (3)



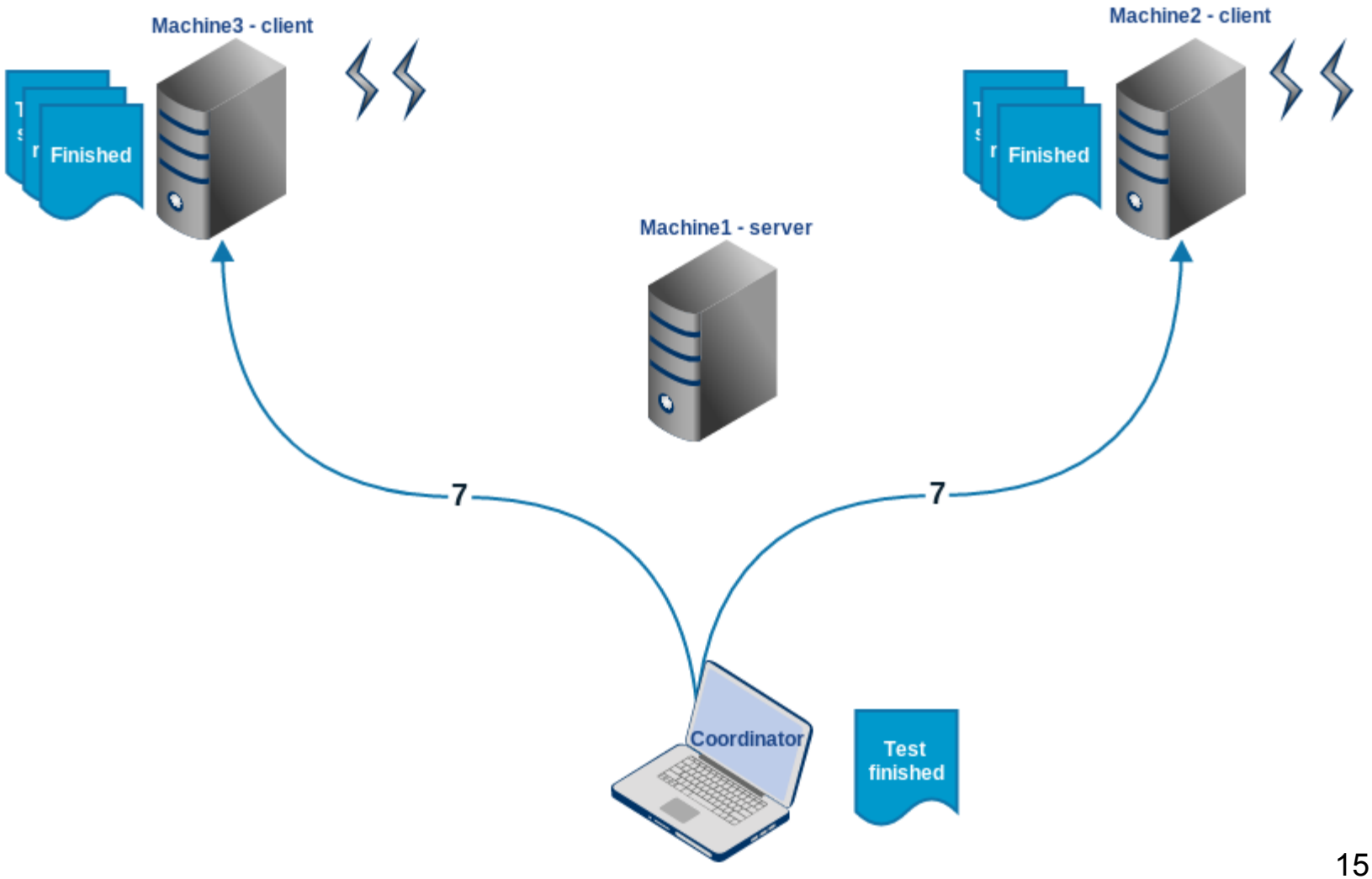
Testing tool - requests finished (4)



Testing tool - requests finished (4)



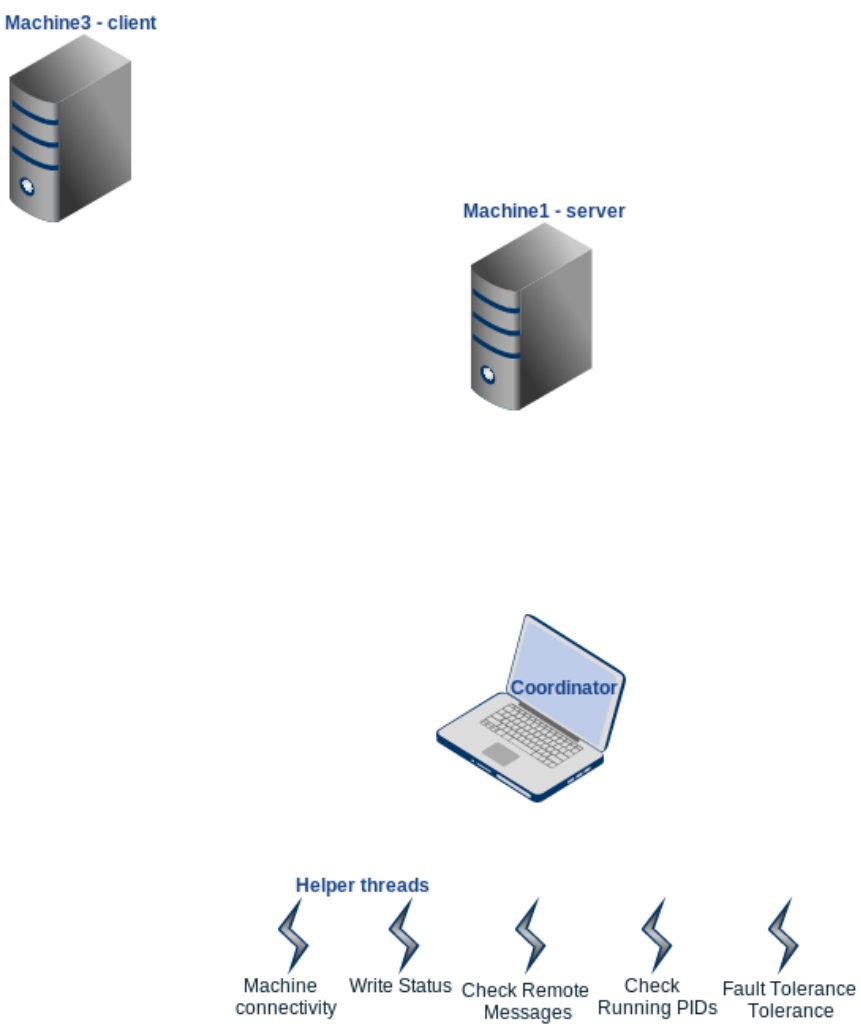
Testing tool - test finished (5)



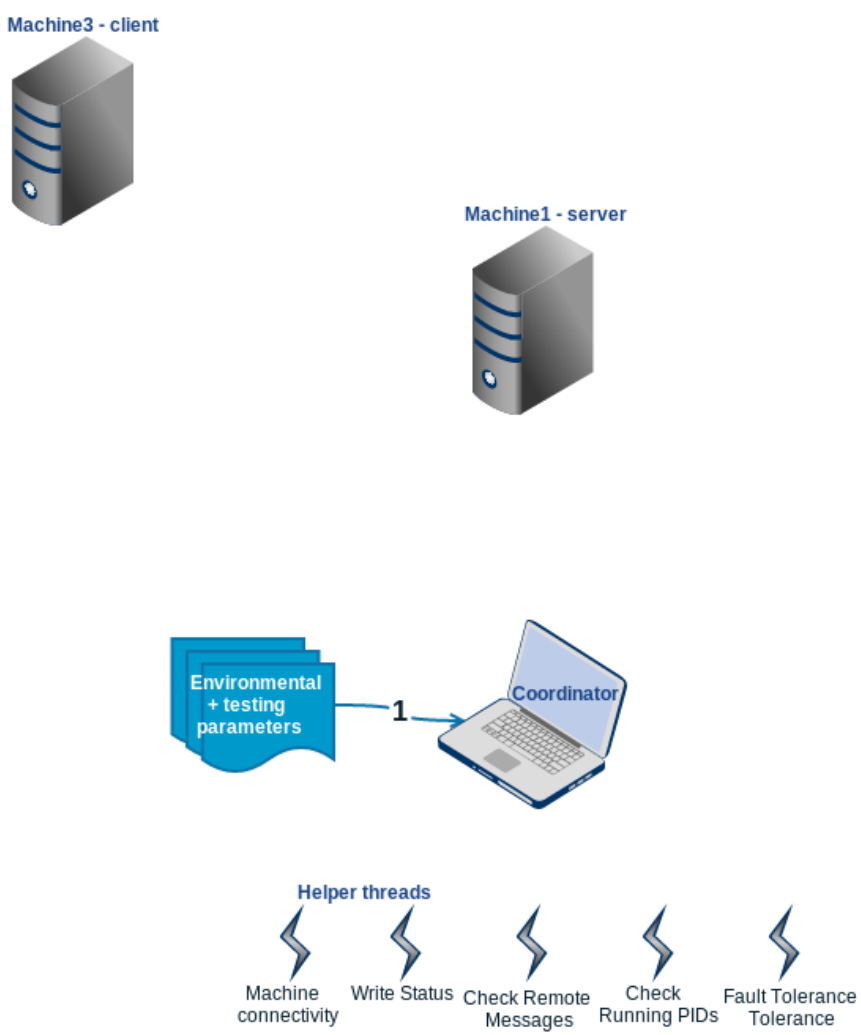
Testing tool - architecture

- Uses .properties files for config purposes
  - Java library: Apache Common configuration
- Coordinates remote server and clients
  - via SSHJ Java library
  - key-based authentication
- Checks the running status and log
- Gathers the results

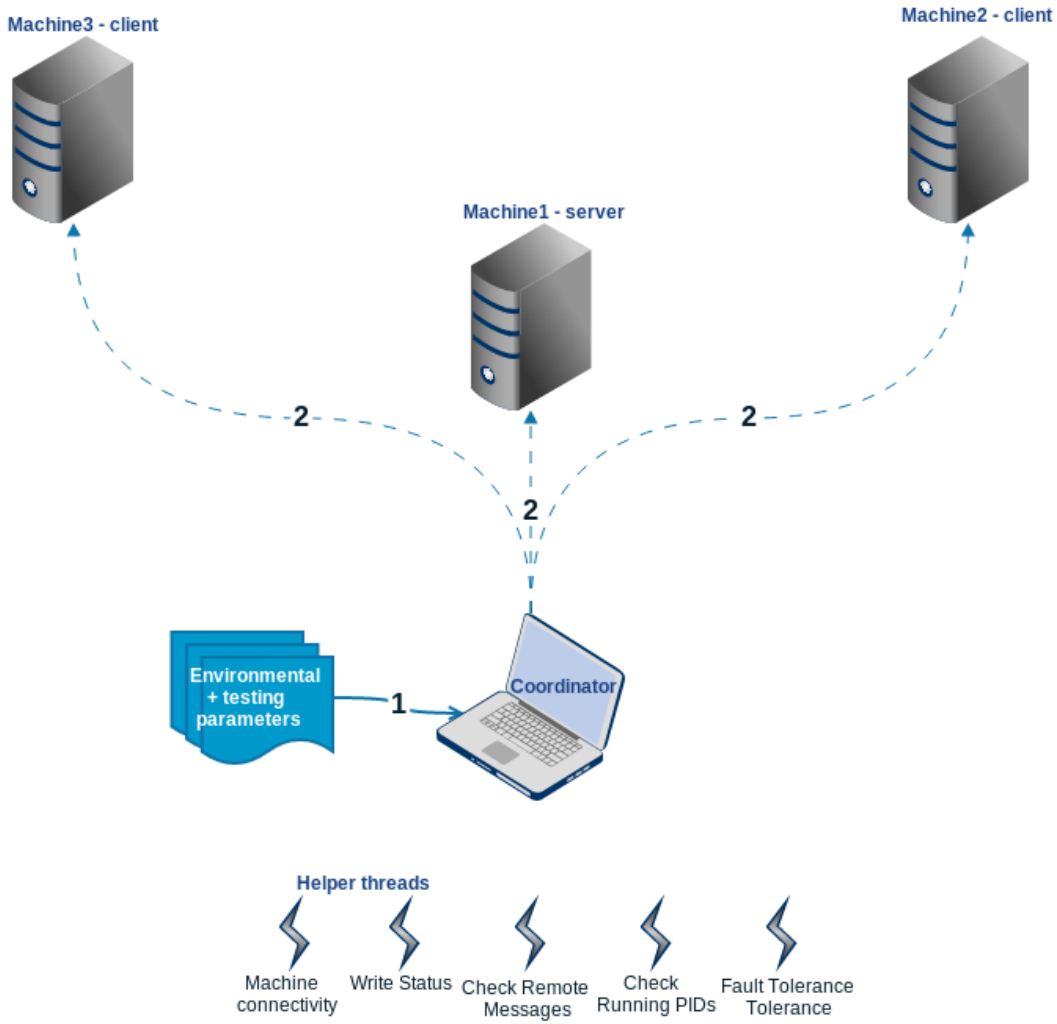
Testing tool - helper threads (0)



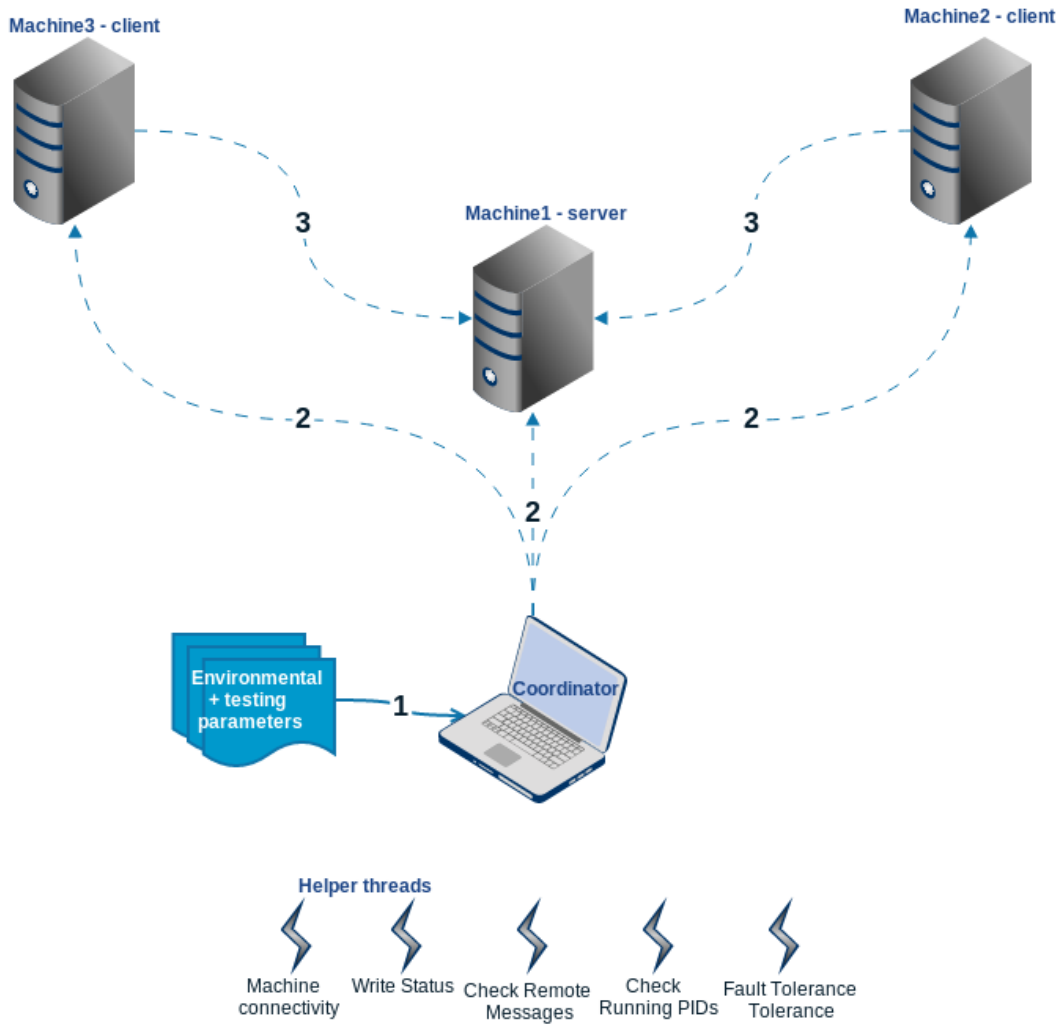
Testing tool - parsing .properties files (1)



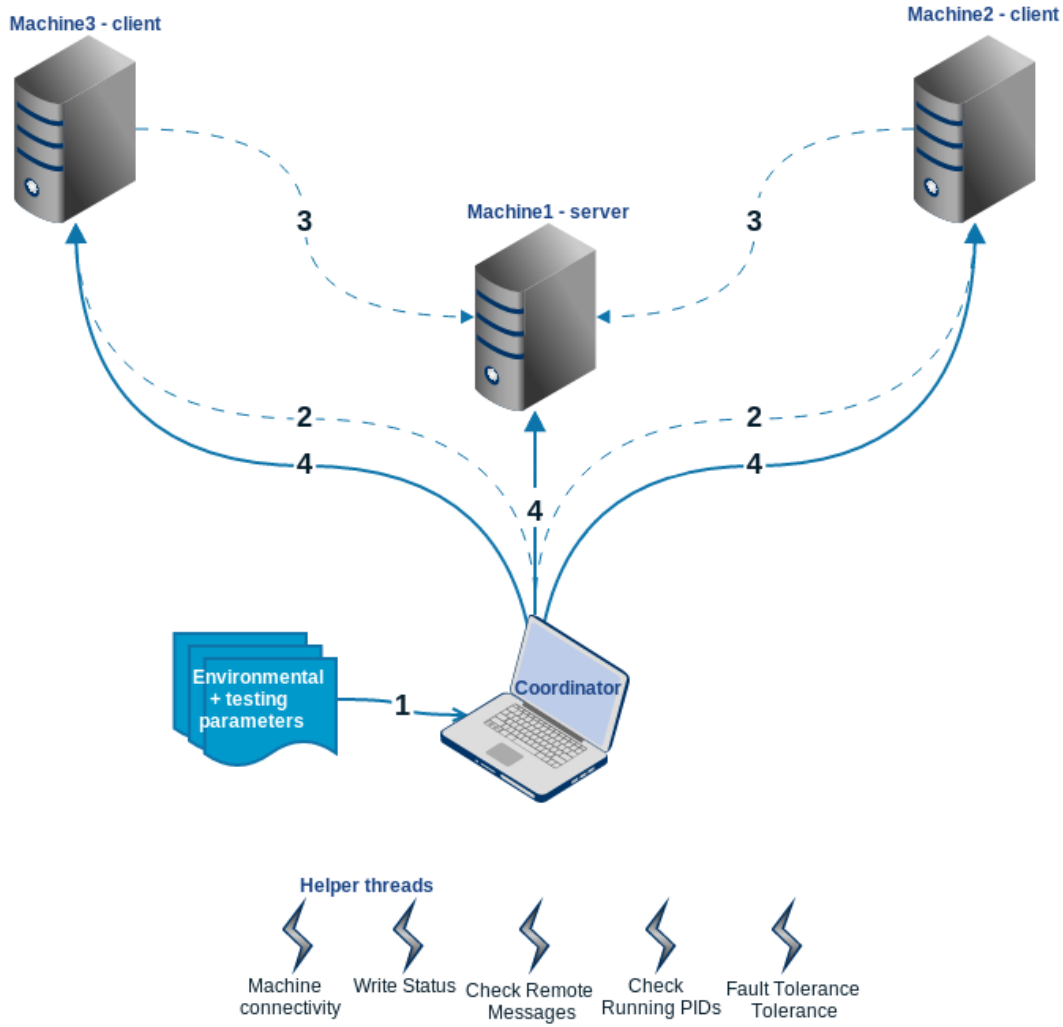
Testing tool - create SSH clients (2)



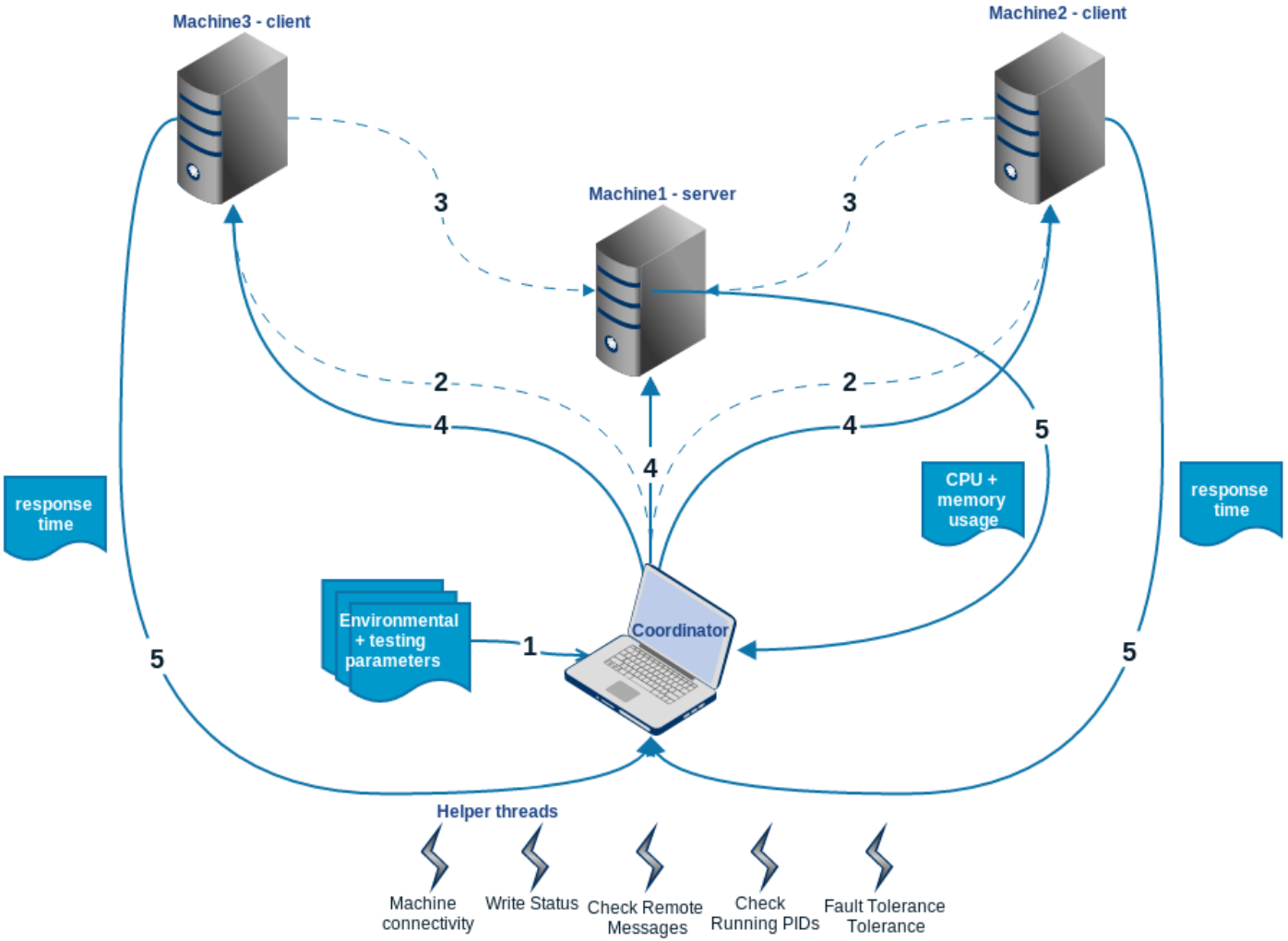
Testing tool - check network connectivity (3)



Testing tool - upload and start programs (4)



Testing tool - collect results (5)



Testing tool - test scenario

##tests to perform by clients for nio2 server type

server type

server.type = nio2

#server mode async or/and sync

server.mode = sync async

#number of times to repeat a test

test.num = 1

#number of threads

test.threads.num = 100

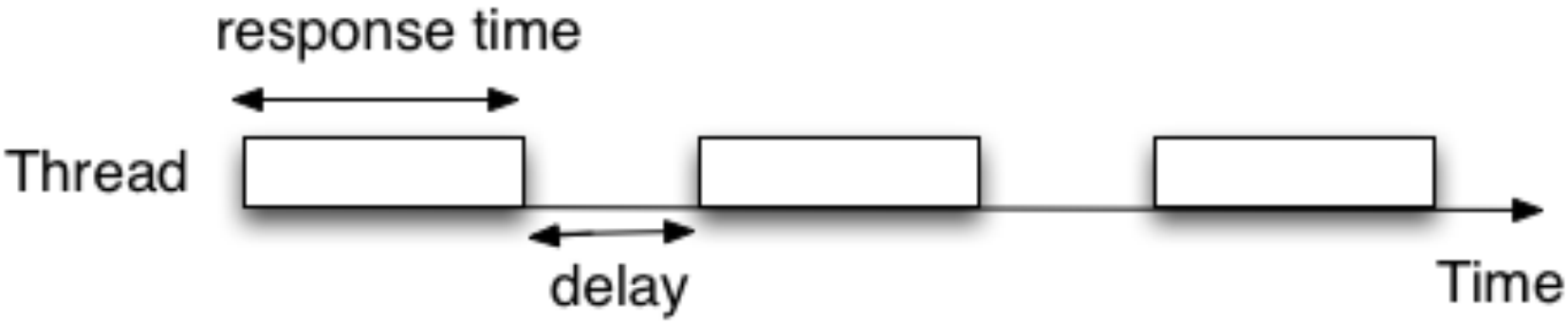
#number of requests, multiple values

test.request.num = 100000

#delays, multiple values

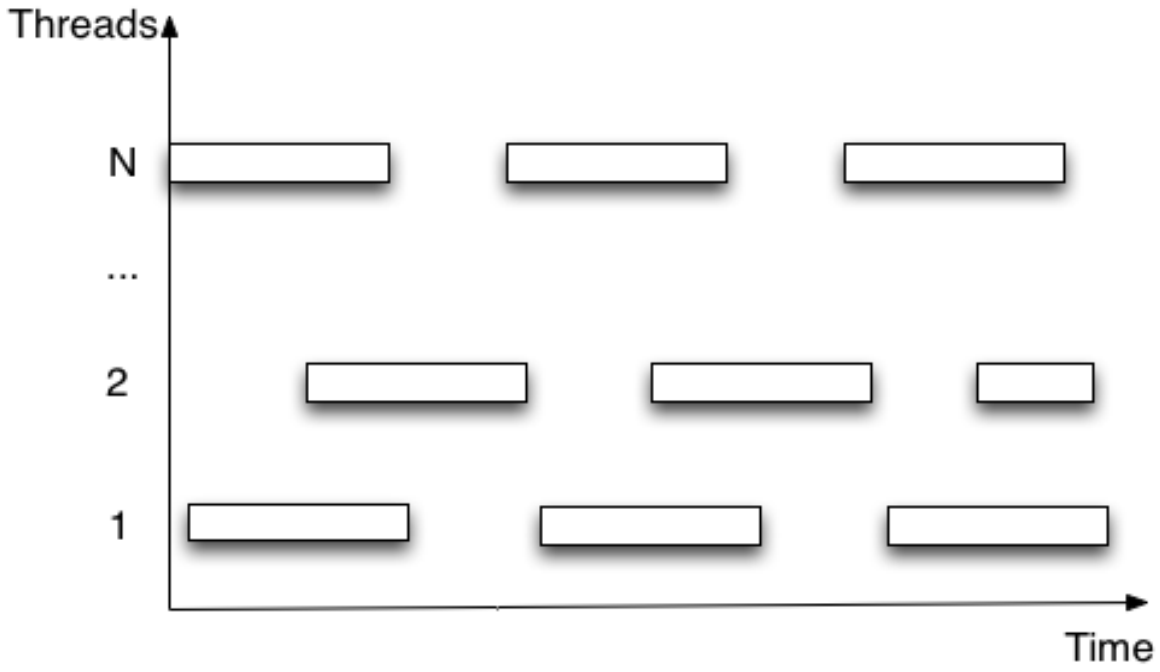
test.delays = 250 225 200 175 150 125 100 75

Data processing



Each client thread makes ~100K requests
Number of client threads is 1000

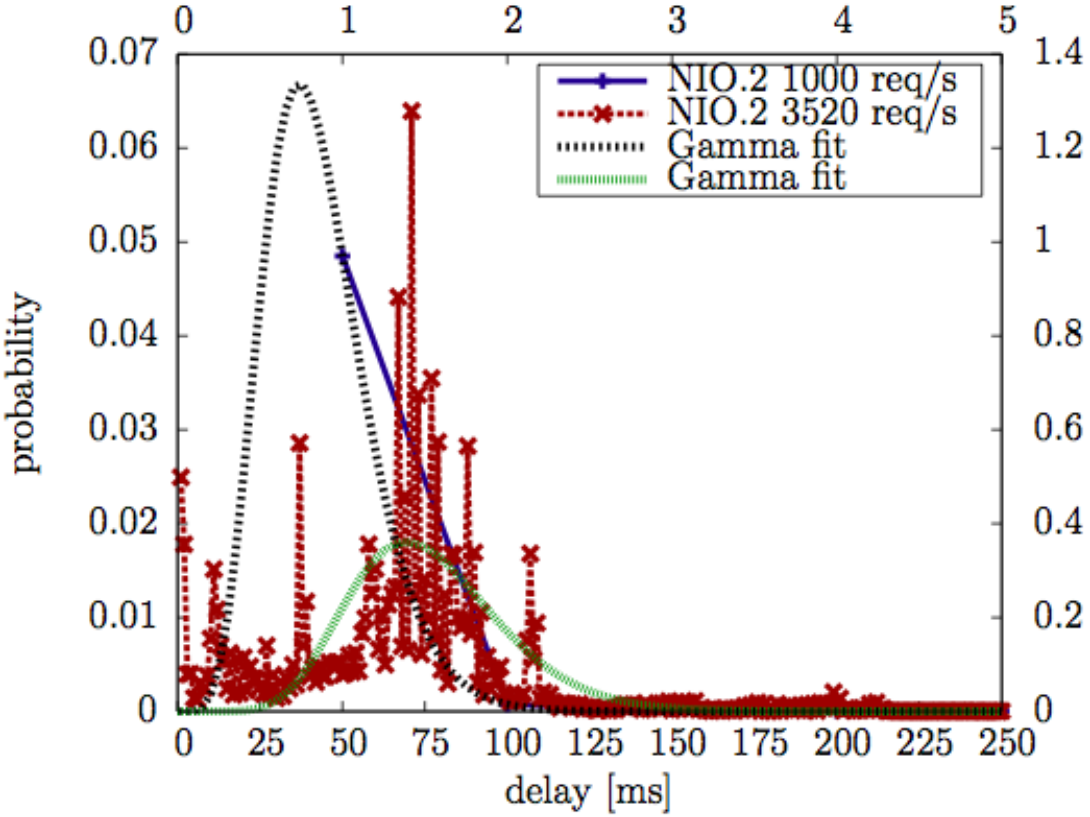
Data processing



Request load = (time)/request number

Previous results

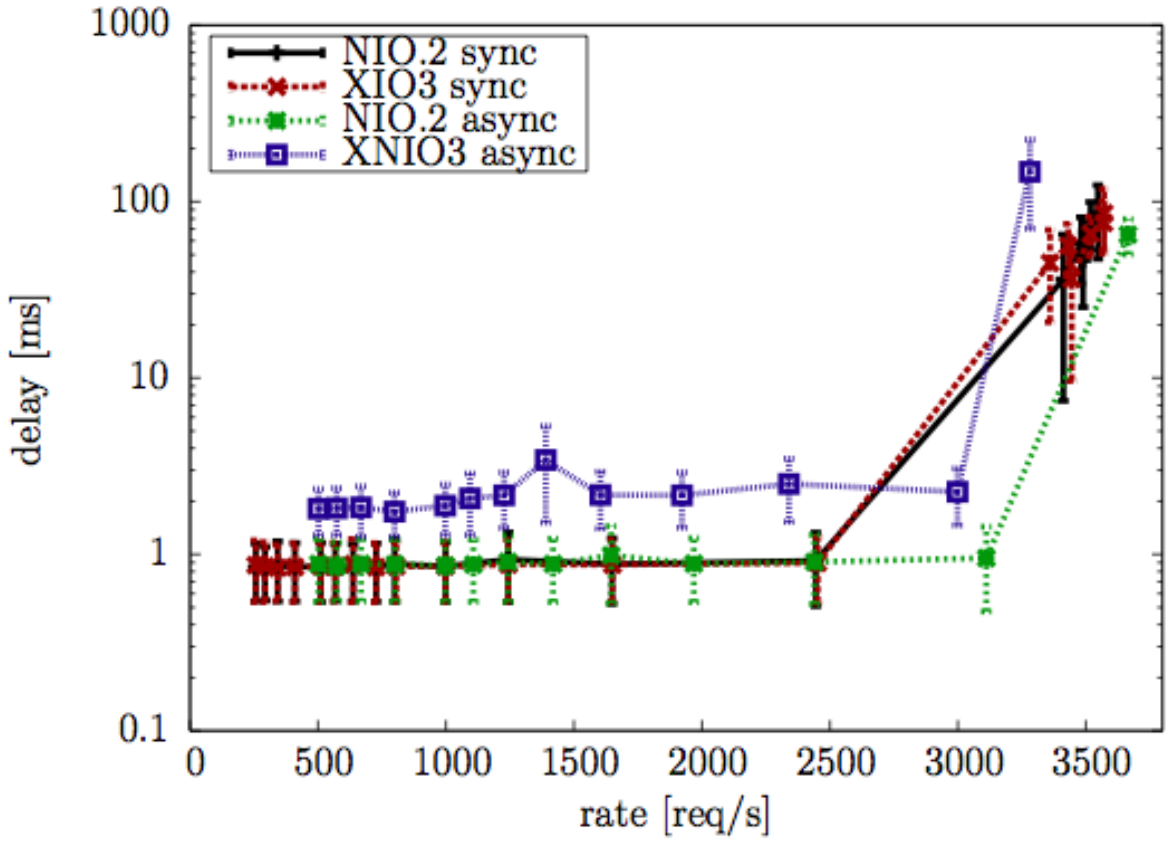
Delay distribution on the example NIO2 synchronous tests.



Gamma distribution is good low request rates

Previous results

Delay against load in NIO2, XNIO3 tests



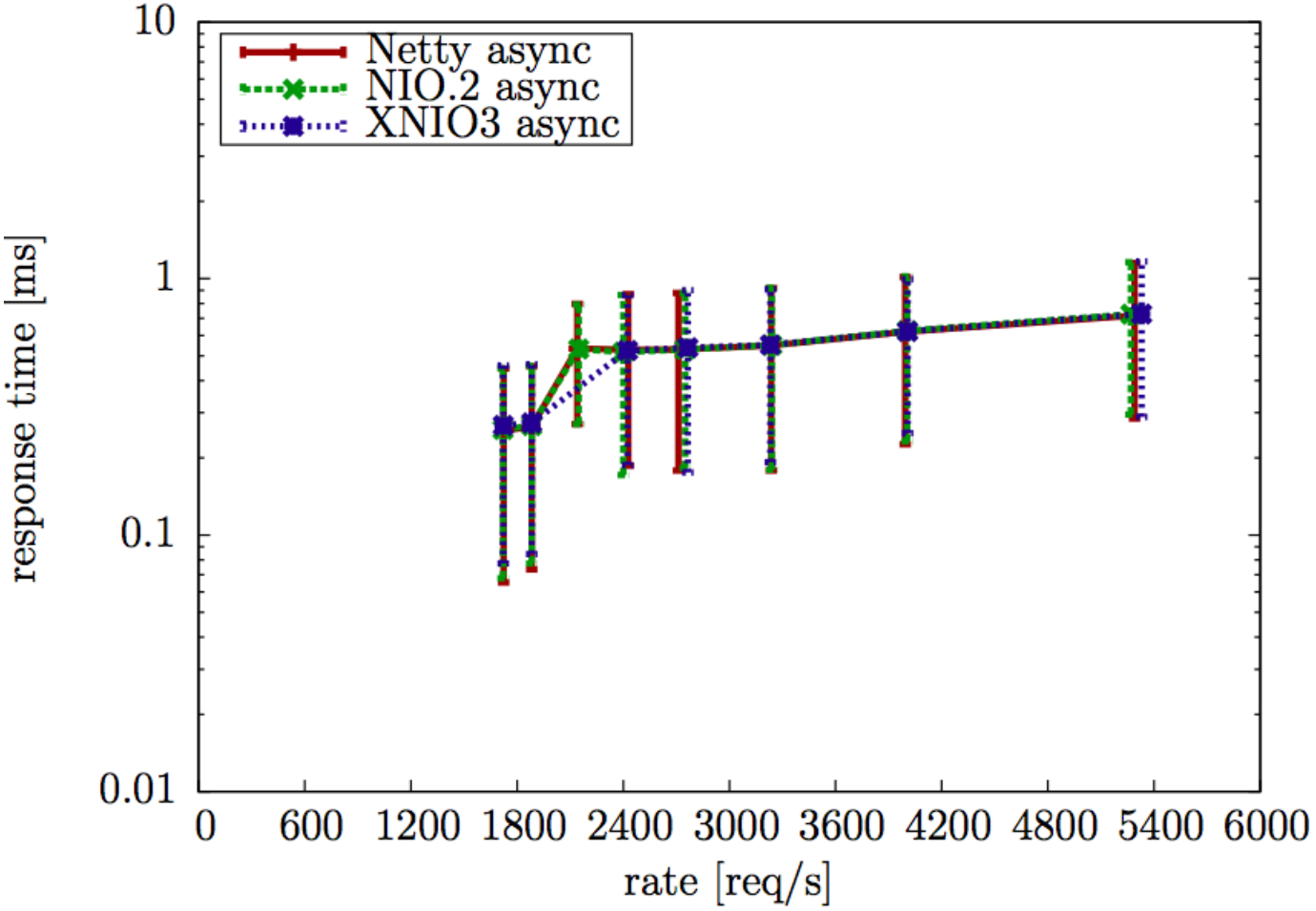
Test conditions

- Client and server machines: 64bit, 8-core, x86_64 3GHz processor, 4GB of available memory
- 1 client with 1000 threads issuing 100K requests
- Two modes: asynchronous and synchronous
- Interested in high request rates. Vary delay between requests (time in ms):

  250, 225, 200, 175, 150, 125, 75

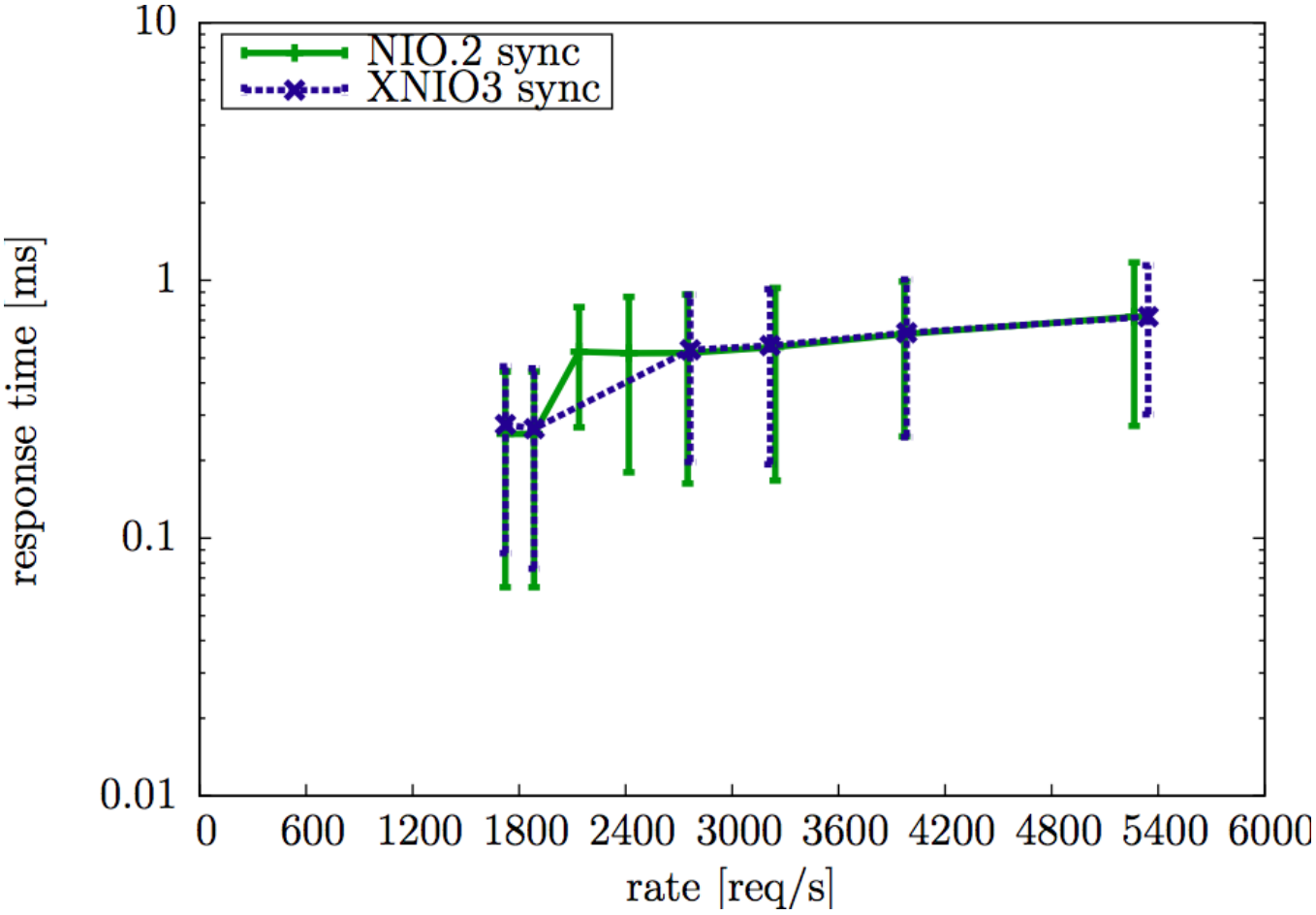
Results

Response time to load for NIO.2, XNIO.3 and Netty.



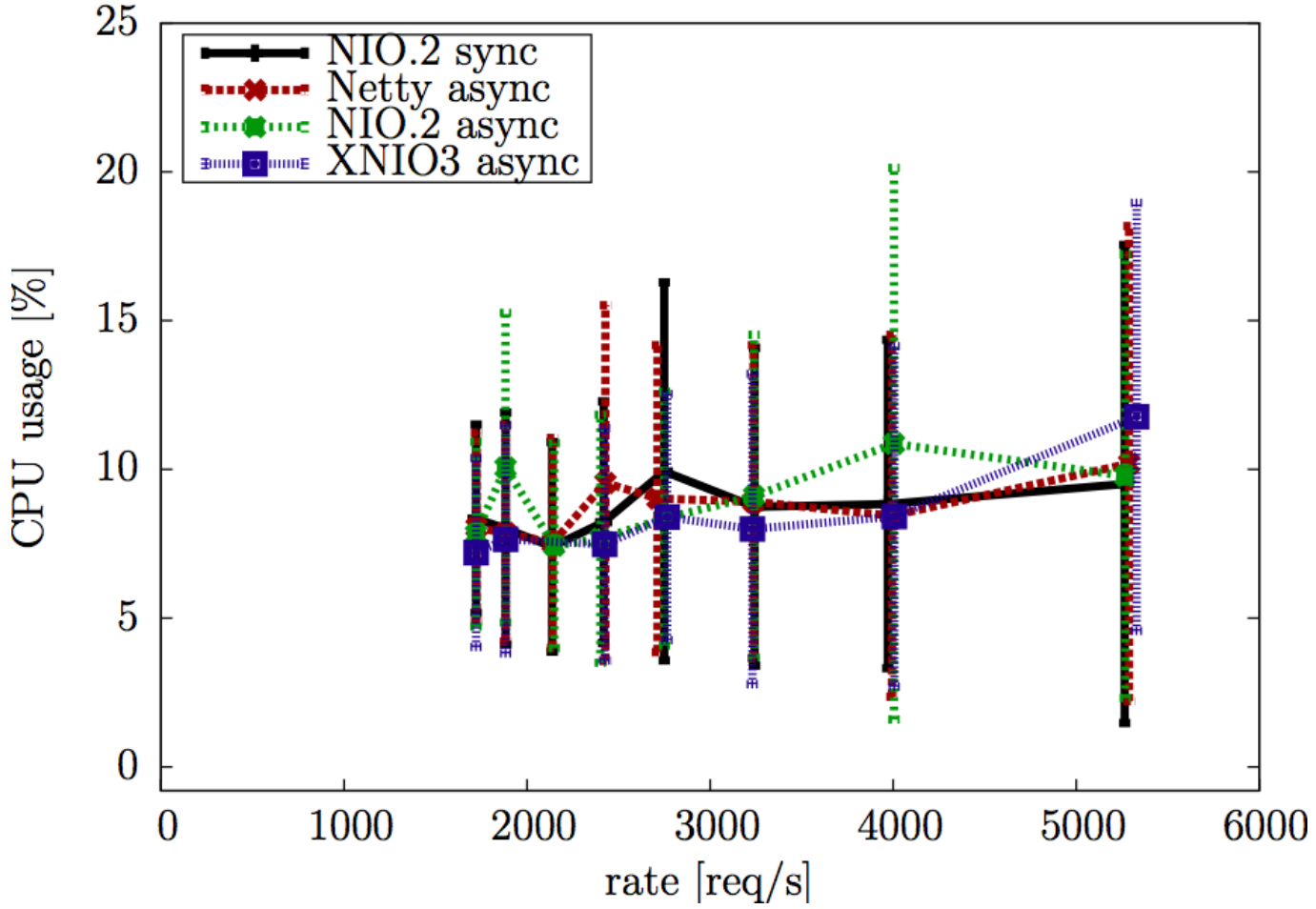
Results

Response time to load for NIO.2, XNIO.3, sync. mode.



Results

CPU consumption for NIO.2, XNIO.3, Netty



Conclusion

- Succeeded to implement flexible tool
  - hopefully not with too many bugs
- Available on github:
  - https://github.com/teodormacicas/jboss_benchmarkIO
- Time restrictions to run more tests and to get meaningful results
- Gained experience and had fun working on open-source project.

Thank you!

Do you have questions?