

---

# Continuous Performance Testing

Confoo.ca

Kore Nordmann (@koredn)

27. Feb 2013

# About me

---

- ▶ Degree in computer science

# About me

---

- ▶ Degree in computer science
- ▶ Professional PHP since 2000

# About me

---

- ▶ Degree in computer science
- ▶ Professional PHP since 2000
- ▶ Open source enthusiast

# About me

---

- ▶ Degree in computer science
- ▶ Professional PHP since 2000
- ▶ Open source enthusiast
- ▶ **Passion for**
  - ▶ Software Design
  - ▶ Automated Testing

# Co-founder of

---



Co-founder of

---



**Helping people to create high quality web applications.**

<http://qafoo.com>

# Co-founder of

---



**Helping people to create high quality web applications.**

<http://qafoo.com>

- ▶ Expert consulting
- ▶ Individual training



# Co-founder of

---



**Helping people to create high quality web applications.**

<http://qafoo.com>

- ▶ Expert consulting
- ▶ Individual training

Get a training on object oriented design for your team!

# Outline

---

Motivation

Conclusion

# Motivation

---

- ▶ Why should we do performance tests?

# Motivation

---

- ▶ Why should we do performance tests?
  - ▶ Locate unknown bottlenecks
  - ▶ Measure behaviour of the full stack

# Motivation

---

- ▶ Why should we do performance tests?
  - ▶ Locate unknown bottlenecks
  - ▶ Measure behaviour of the full stack
- ▶ Why should we do that continuously?

# Motivation

---

- ▶ Why should we do performance tests?
  - ▶ Locate unknown bottlenecks
  - ▶ Measure behaviour of the full stack
- ▶ Why should we do that continuously?
  - ▶ Find performance regressions
  - ▶ Ensure optimizations are persistent

# Often used tools

---

- ▶ Often misused tools

# Often used tools

---

- ▶ Often misused tools
  - ▶ siege
  - ▶ ApacheBench (ab)



# Often used tools

---

- ▶ Often misused tools
  - ▶ siege
  - ▶ ApacheBench (ab)
- ▶ Testing for micro-optimizations

# Often used tools

---

- ▶ Often misused tools
  - ▶ siege
  - ▶ ApacheBench (ab)
- ▶ Testing for micro-optimizations
  - ▶ Evaluating Hello-World-examples of Frameworks

# Often used tools

---

- ▶ Often misused tools
  - ▶ siege
  - ▶ ApacheBench (ab)
- ▶ Testing for micro-optimizations
  - ▶ Evaluating Hello-World-examples of Frameworks
- ▶ Useful tools, I won't talk about
  - ▶ xDebug Profiling
  - ▶ xhProf
  - ▶ Database profiling tools
  - ▶ System metrics (Graphite, ...)

# The problem is more complex

---

- ▶ Your task: Create a new webshop

# The problem is more complex

---


- ▶ Your task: Create a new webshop
  - ▶ Assume it's march

# The problem is more complex

---


- ▶ Your task: Create a new webshop
  - ▶ Assume it's march
  - ▶ The deadline is October this year, right before Christmas

## Awesome Shop

 0 articles  
0.00 €

### Smartphone

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.



1337,-- €  
5 items in stock

#### Comments




● ● ● ● ● ● ● ● ● ●

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor.


# A webshop

## Awesome Shop

 2 articles  
42.32 €


### Smartphone

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.



1337,-- €  
5 items in stock

#### Comments




● ● ● ● ● ● ● ●

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor.




# A webshop

## Awesome Shop

 2 articles  
42.32 €


### Smartphone

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.



1337,-- €  
5 items in stock

#### Comments




5 ●●●●●●●●

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor.


# A webshop

## Awesome Shop

 2 articles  
42.32 €


### Smartphone

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.



1337,- €  
5 items in stock

#### Comments




● ● ● ● ● ● ● ●

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor.


# A webshop

## Awesome Shop

 2 articles  
42.32 €


### Smartphone

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.



1337,-- €  
5 items in stock

#### Comments



● ● ● ● ● ● ● ●

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor.

# A real tool for performance tests

---

- ▶ JMeter

# A real tool for performance tests

---

- ▶ JMeter
  - ▶ Complex user paths

# A real tool for performance tests

---

- ▶ **JMeter**
  - ▶ Complex user paths
  - ▶ Concurrent requests

# A real tool for performance tests

---

- ▶ **JMeter**
  - ▶ Complex user paths
  - ▶ Concurrent requests
  - ▶ Record on proxy

# A real tool for performance tests

---

- ▶ **JMeter**
  - ▶ Complex user paths
  - ▶ Concurrent requests
  - ▶ Record on proxy
  - ▶ **Clustering**



# JMeter 101

---

- ▶ Thread Group
  - ▶ An execution plan (like a user registration)

# JMeter 101

---

- ▶ Thread Group
  - ▶ An execution plan (like a user registration)
- ▶ Controller
  - ▶ Controls how samplers are executed (loop, random, ...)

# JMeter 101

---

- ▶ Thread Group
  - ▶ An execution plan (like a user registration)
- ▶ Controller
  - ▶ Controls how samplers are executed (loop, random, ...)
- ▶ Config Element
  - ▶ Configuration, optionally from external sources

# JMeter 101

---

- ▶ Thread Group
  - ▶ An execution plan (like a user registration)
- ▶ Controller
  - ▶ Controls how samplers are executed (loop, random, ...)
- ▶ Config Element
  - ▶ Configuration, optionally from external sources
- ▶ Timer
  - ▶ Defining timing constraints for sampler execution

# JMeter 101

---

- ▶ Thread Group
  - ▶ An execution plan (like a user registration)
- ▶ Controller
  - ▶ Controls how samplers are executed (loop, random, ...)
- ▶ Config Element
  - ▶ Configuration, optionally from external sources
- ▶ Timer
  - ▶ Defining timing constraints for sampler execution
- ▶ Sampler
  - ▶ Perform the actual work (like HTTP requests)

# Getting started

---

- ▶ Create a test plan
  - ▶ What do users actually do on your site?

# Getting started

---

- ▶ Create a test plan
  - ▶ What do users actually do on your site?
- ▶ Example:
  - ▶ Random browser
  - ▶ User registration
  - ▶ Sign on
  - ▶ Shopping with checkout
  - ▶ Commenting products

# Using JMeter

---

Demo



# Automation

---

- ▶ Ant JMeter integration

# Automation

---

- ▶ Ant JMeter integration
- ▶ **Automation of your environment**

# Automation

---

- ▶ Ant JMeter integration
- ▶ Automation of your environment
  - ▶ Setting up different software versions

# Automation

---

- ▶ Ant JMeter integration
- ▶ **Automation of your environment**
  - ▶ Setting up different software versions
  - ▶ Testing with different extensions

# Automation

---

- ▶ Ant JMeter integration
- ▶ **Automation of your environment**
  - ▶ Setting up different software versions
  - ▶ Testing with different extensions
  - ▶ Running different database setups

# Automation

---

- ▶ Ant JMeter integration
- ▶ **Automation of your environment**
  - ▶ Setting up different software versions
  - ▶ Testing with different extensions
  - ▶ Running different database setups
  - ▶ **Handling multi-node environments**

# Automation

---

- ▶ Ant JMeter integration
- ▶ **Automation of your environment**
  - ▶ Setting up different software versions
  - ▶ Testing with different extensions
  - ▶ Running different database setups
  - ▶ Handling multi-node environments
  - ▶ **Plain Ant or Puppet, Chef, Vagrant**

# Apache Ant example

---

```
26 <target name="build-apc-bytecode-cache-user-cache-file"  
27     depends="setup-apc-bytecode-file-cache, _build"  
28     description=">_Run_with_APC_opcode_cache_and_file_based_cache." />  
29  
30 <target name="build-apc-bytecode-cache-user-cache-apc"  
31     depends="setup-apc-bytecode-apc-cache, _build"  
32     description=">_Run_with_APC_opcode_cache_and_APC_based_cache." />  
  
211 <target name="-remote-exec-parallel">  
212     <subant target="{target}" inheritall="true">  
213         <fileset dir="{project.dir}" includes="server*.xml" />  
214     </subant>  
215 </target>  
216  
217 <target name="-remote-exec">  
218     <sshexec command="{command}"  
219         username="{ssh.username}"  
220         password="{ssh.password}"  
221         host="{hostname}"  
222         trust="true" />  
223 </target>
```



# Apache Ant example

```
26 <target name="build-apc-bytecode-cache-user-cache-file"  
27     depends="setup-apc-bytecode-file-cache, _build"  
28     description="->_Run_with_APC_opcode_cache_and_file_based_cache." />  
29  
30 <target name="build-apc-bytecode-cache-user-cache-apc"  
31     depends="setup-apc-bytecode-apc-cache, _build"  
32     description="->_Run_with_APC_opcode_cache_and_APC_based_cache." />  
  
211 <target name="-remote-exec-parallel">  
212     <subant target="${target}" inheritall="true">  
213         <fileset dir="${project.dir}" includes="server*.xml" />  
214     </subant>  
215 </target>  
216  
217 <target name="-remote-exec">  
218     <sshexec command="${command}"  
219         username="${ssh.username}"  
220         password="${ssh.password}"  
221         host="${hostname}"  
222         trust="true" />  
223 </target>
```

# Apache Ant example

```
26 <target name="build-apc-bytecode-cache-user-cache-file"  
27     depends="setup-apc-bytecode-file-cache, _build"  
28     description="Run with APC opcode cache and file based cache." />  
29  
30 <target name="build-apc-bytecode-cache-user-cache-apc"  
31     depends="setup-apc-bytecode-apc-cache, _build"  
32     description="Run with APC opcode cache and APC based cache." />  
  
211 <target name="-remote-exec-parallel">  
212     <subant target="${target}" inheritall="true">  
213         <fileset dir="${project.dir}" includes="server*.xml" />  
214     </subant>  
215 </target>  
  
216  
217 <target name="-remote-exec">  
218     <sshexec command="${command}"  
219         username="${ssh.username}"  
220         password="${ssh.password}"  
221         host="${hostname}"  
222         trust="true" />  
223 </target>
```

# Apache Ant example

```
26 <target name="build-apc-bytecode-cache-user-cache-file"  
27     depends="setup-apc-bytecode-file-cache, _build"  
28     description=">_Run_with_APC_opcode_cache_and_file_based_cache." />  
29  
30 <target name="build-apc-bytecode-cache-user-cache-apc"  
31     depends="setup-apc-bytecode-apc-cache, _build"  
32     description=">_Run_with_APC_opcode_cache_and_APC_based_cache." />
```

```
211 <target name="-remote-exec-parallel">  
212     <subant target="${target}" inheritall="true">  
213         <fileset dir="${project.dir}" includes="server*.xml" />  
214     </subant>  
215 </target>  
216  
217 <target name="-remote-exec">  
218     <sshexec command="${command}"  
219         username="${ssh.username}"  
220         password="${ssh.password}"  
221         host="${hostname}"  
222         trust="true" />  
223 </target>
```

# Apache Ant example

```
26 <target name="build-apc-bytecode-cache-user-cache-file"
27     depends="setup-apc-bytecode-file-cache, build"
28     description="Run with APC opcode cache and file based cache." />
29
30 <target name="build-apc-bytecode-cache-user-cache-apc"
31     depends="setup-apc-bytecode-apc-cache, build"
32     description="Run with APC opcode cache and APC based cache." />
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111 <target name="-remote-exec-parallel">
112     <subant target="{target}" inheritall="true">
113         <fileset dir="{project.dir}" includes="server*.xml" />
114     </subant>
115 </target>
116
117 <target name="-remote-exec">
118     <sshexec command="{command}"
119         username="{ssh.username}"
120         password="{ssh.password}"
121         host="{hostname}"
122         trust="true" />
123 </target>
```

# Apache Ant example

---

```
454 <target name="-restart-host">
455     <antcall target="-remote-exec">
456         <param name="command" value="shutdown_r_now" />
457     </antcall>
458
459     <echo taskname="waitfor" message="Wait_for_${hostname}_to_stop ..." />
460     <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
461         <not>
462             <http url="http://${hostname}"/>
463         </not>
464     </waitfor>
465
466     <echo taskname="waitfor" message="Wait_for_${hostname}_is_up_again ..." />
467     <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
468         <http url="http://${hostname}"/>
469     </waitfor>
470 </target>
```

# Apache Ant example

---

```
454 <target name="-restart-host">
455   <antcall target="-remote-exec">
456     <param name="command" value="shutdown_r_now" />
457   </antcall>
458
459   <echo taskname="waitfor" message="Wait_for_${hostname}_to_stop ..." />
460   <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
461     <not>
462       <http url="http://${hostname}"/>
463     </not>
464   </waitfor>
465
466   <echo taskname="waitfor" message="Wait_for_${hostname}_is_up_again ..." />
467   <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
468     <http url="http://${hostname}"/>
469   </waitfor>
470 </target>
```

# Apache Ant example

```
454 <target name="-restart-host">
455     <antcall target="-remote-exec">
456         <param name="command" value="shutdown_r_now" />
457     </antcall>
458
459     <echo taskname="waitfor" message="Wait_for_${hostname}_to_stop..." />
460     <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
461         <not>
462             <http url="http://${hostname}"/>
463         </not>
464     </waitfor>
465
466     <echo taskname="waitfor" message="Wait_for_${hostname}_is_up_again..." />
467     <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
468         <http url="http://${hostname}"/>
469     </waitfor>
470 </target>
```

# Apache Ant example

---

```
454 <target name="-restart-host">
455   <antcall target="-remote-exec">
456     <param name="command" value="shutdown -r now" />
457   </antcall>
458
459   <echo taskname="waitfor" message="Wait for ${hostname} to stop ..." />
460   <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
461     <not>
462       <http url="http://${hostname}"/>
463     </not>
464   </waitfor>
465
466   <echo taskname="waitfor" message="Wait for ${hostname} is up again ..." />
467   <waitfor maxwait="5" maxwaitunit="minute" checkevery="100">
468     <http url="http://${hostname}"/>
469   </waitfor>
470 </target>
```



# Apache Ant example

---

```
421 <target name="jmeter" depends="-settings-init,-start-jmeter" />
422
423 <target name="-start-jmeter">
424   <antcall target="-start-jmeter-before-hook" />
425
426   <jmeter jmeterhome="${local.jmeter.home.dir}"
427     resultlog="${local.jmeter.log.file}"
428     testplan="${local.jmeter.test.dir}/${jmeter.file}">
429
430     <property name="jmeter.data.dir" value="${local.project.data.dir}" />
431     <property name="jmeter.rampup.time" value="${jmeter.rampup.time}" />
432     <property name="jmeter.execution.time" value="${jmeter.execution.time}" />
433   </jmeter>
434
435   <antcall target="-start-jmeter-after-hook" />
436 </target>
```

# Apache Ant example

---

```
421 <target name="jmeter" depends="-settings-init,-start-jmeter" />
422
423 <target name="-start-jmeter">
424   <antcall target="-start-jmeter-before-hook" />
425
426   <jmeter jmeterhome="${local.jmeter.home.dir}"
427     resultlog="${local.jmeter.log.file}"
428     testplan="${local.jmeter.test.dir}/${jmeter.file}">
429
430     <property name="jmeter.data.dir" value="${local.project.data.dir}" />
431     <property name="jmeter.rampup.time" value="${jmeter.rampup.time}" />
432     <property name="jmeter.execution.time" value="${jmeter.execution.time}" />
433   </jmeter>
434
435   <antcall target="-start-jmeter-after-hook" />
436 </target>
```

# Apache Ant example

---

```
421 <target name="jmeter" depends="-settings-init,-start-jmeter" />
422
423 <target name="-start-jmeter">
424   <antcall target="-start-jmeter-before-hook" />
425
426   <jmeter jmeterhome="${local.jmeter.home.dir}"
427     resultlog="${local.jmeter.log.file}"
428     testplan="${local.jmeter.test.dir}/${jmeter.file}">
429
430     <property name="jmeter.data.dir" value="${local.project.data.dir}" />
431     <property name="jmeter.rampup.time" value="${jmeter.rampup.time}" />
432     <property name="jmeter.execution.time" value="${jmeter.execution.time}" />
433   </jmeter>
434
435   <antcall target="-start-jmeter-after-hook" />
436 </target>
```

# Apache Ant example

```
351 <target name="-backup-data-from-host">
352   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      error.log}'"
353     localtofile="${local.builddir}/${hostname}-error.log"
354     trust="true" />
355
356   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      access.log}'"
357     localtofile="${local.builddir}/${hostname}-access.log"
358     trust="true" />
359
360   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.php.error.
      log}'"
361     localtofile="${local.builddir}/${hostname}-php_errors.log"
362     trust="true" />
363
364   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.basedir}/
      scripts/load.log'"
365     localtofile="${local.builddir}/${hostname}-load.log"
366     trust="true" />
367
368 </target>
```

# Apache Ant example

```
351 <target name="-backup-data-from-host">
352   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
353     error.log}'"
354     localtofile="${local.builddir}/${hostname}-error.log"
355     trust="true" />
356
357   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
358     access.log}'"
359     localtofile="${local.builddir}/${hostname}-access.log"
360     trust="true" />
361
362   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.php.error.
363     log}'"
364     localtofile="${local.builddir}/${hostname}-php_errors.log"
365     trust="true" />
366
367   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.basedir}/
368     scripts/load.log'"
369     localtofile="${local.builddir}/${hostname}-load.log"
370     trust="true" />
371 </target>
```

# Apache Ant example

```
351 <target name="-backup-data-from-host">
352   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      error.log}'"
353     localtofile="${local.builddir}/${hostname}-error.log"
354     trust="true" />
355
356   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      access.log}'"
357     localtofile="${local.builddir}/${hostname}-access.log"
358     trust="true" />
359
360   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.php.error.
      log}'"
361     localtofile="${local.builddir}/${hostname}-php_errors.log"
362     trust="true" />
363
364   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.basedir}/
      scripts/load.log'"
365     localtofile="${local.builddir}/${hostname}-load.log"
366     trust="true" />
367
368 </target>
```

# Apache Ant example

```
351 <target name="-backup-data-from-host">
352   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      error.log}'"
353     localtofile="${local.builddir}/${hostname}-error.log"
354     trust="true" />
355
356   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      access.log}'"
357     localtofile="${local.builddir}/${hostname}-access.log"
358     trust="true" />
359
360   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.php.error.
      log}'"
361     localtofile="${local.builddir}/${hostname}-php_errors.log"
362     trust="true" />
363
364   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.basedir}/
      scripts/load.log'"
365     localtofile="${local.builddir}/${hostname}-load.log"
366     trust="true" />
367
368 </target>
```

# Apache Ant example

```
351 <target name="-backup-data-from-host">
352   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      error.log}'"
353     localtofile="${local.builddir}/${hostname}-error.log"
354     trust="true" />
355
356   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.webserver.
      access.log}'"
357     localtofile="${local.builddir}/${hostname}-access.log"
358     trust="true" />
359
360   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.php.error.
      log}'"
361     localtofile="${local.builddir}/${hostname}-php_errors.log"
362     trust="true" />
363
364   <scp remotefile="${ssh.username}:${ssh.password}@${hostname}:'${remote.basedir}/
      scripts/load.log'"
365     localtofile="${local.builddir}/${hostname}-load.log"
366     trust="true" />
367
368 </target>
```



What do we actually test?

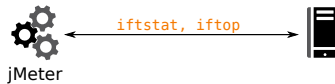
# Test Setup

---

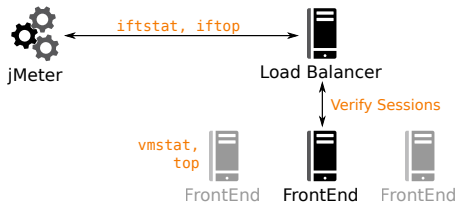


# Test Setup

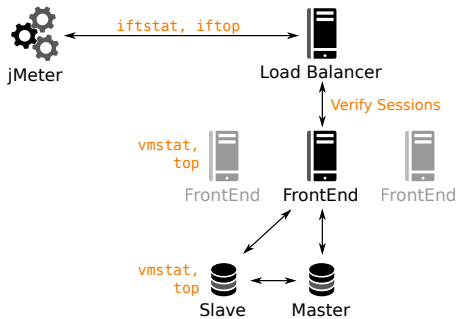
---



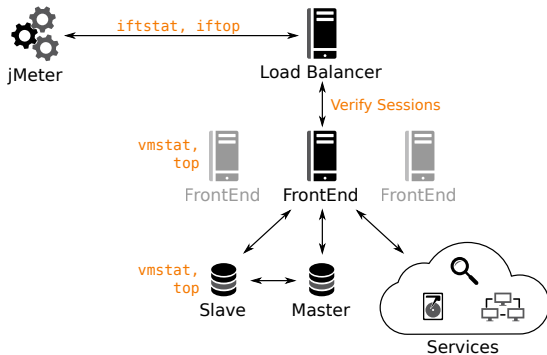
# Test Setup



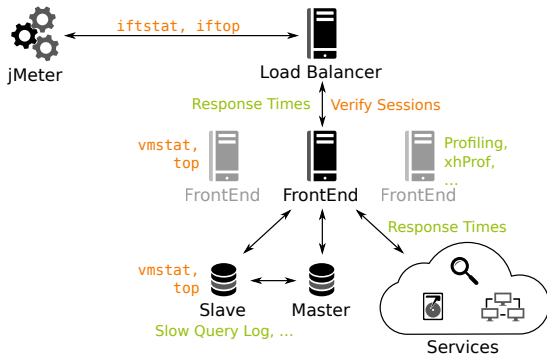
# Test Setup



# Test Setup



# Test Setup



# Hardware

---

- ▶ Test in a realistic environment



# Hardware

---

- ▶ Test in a realistic environment
  - ▶ If your software runs in the cloud test against virtual environment
  - ▶ If you use real hardware, also test against real hardware

# Hardware

---

- ▶ Test in a realistic environment
  - ▶ If your software runs in the cloud test against virtual environment
  - ▶ If you use real hardware, also test against real hardware
- ▶ **JMeter** might have serious hardware requirements

# Hardware

---

- ▶ Test in a realistic environment
  - ▶ If your software runs in the cloud test against virtual environment
  - ▶ If you use real hardware, also test against real hardware
- ▶ JMeter might have serious hardware requirements
  - ▶ Use real hardware
  - ▶ Use the biggest VM available
  - ▶ Ensure that not the JMeter hardware is the bottleneck

# Hardware

---

- ▶ Test in a realistic environment
  - ▶ If your software runs in the cloud test against virtual environment
  - ▶ If you use real hardware, also test against real hardware
- ▶ JMeter might have serious hardware requirements
  - ▶ Use real hardware
  - ▶ Use the biggest VM available
  - ▶ Ensure that not the JMeter hardware is the bottleneck
- ▶ Be sure that the network is not the bottleneck

# Hardware

---

- ▶ Test in a realistic environment
  - ▶ If your software runs in the cloud test against virtual environment
  - ▶ If you use real hardware, also test against real hardware
- ▶ JMeter might have serious hardware requirements
  - ▶ Use real hardware
  - ▶ Use the biggest VM available
  - ▶ Ensure that not the JMeter hardware is the bottleneck
- ▶ Be sure that the network is not the bottleneck
  - ▶ See `ifstat`, `iftop`

# Hardware

---

- ▶ Test in a realistic environment
  - ▶ If your software runs in the cloud test against virtual environment
  - ▶ If you use real hardware, also test against real hardware
- ▶ JMeter might have serious hardware requirements
  - ▶ Use real hardware
  - ▶ Use the biggest VM available
  - ▶ Ensure that not the JMeter hardware is the bottleneck
- ▶ Be sure that the network is not the bottleneck
  - ▶ See `ifstat`, `iftop`
- ▶ Measure several system metrics

# Hardware

---

- ▶ Test in a realistic environment
  - ▶ If your software runs in the cloud test against virtual environment
  - ▶ If you use real hardware, also test against real hardware
- ▶ JMeter might have serious hardware requirements
  - ▶ Use real hardware
  - ▶ Use the biggest VM available
  - ▶ Ensure that not the JMeter hardware is the bottleneck
- ▶ Be sure that the network is not the bottleneck
  - ▶ See `ifstat`, `iftop`
- ▶ Measure several system metrics
  - ▶ See `vmstat`, `top`

# Extended Statistics

---

Demo



# Continuous Performance

---

- ▶ Plugins available for:
  - ▶ Jenkins
  - ▶ Sonar

# Continuous Performance

---

- ▶ Plugins available for:
  - ▶ Jenkins
  - ▶ Sonar
- ▶ Maintaining all those servers can be expensive

# Continuous testing with Jenkins

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

[Performance Trend](#)

**Build History** (trend)

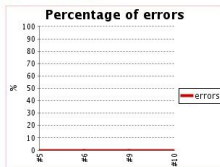
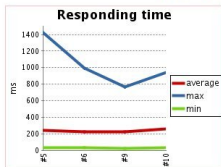
- #10 [Mar 22, 2010 10:36:48 AM](#)
- #9 [Mar 22, 2010 9:59:28 AM](#)
- #8 [Mar 22, 2010 9:46:45 AM](#)
- #7 [Mar 22, 2010 9:38:15 AM](#)
- #6 [Mar 9, 2010 1:22:57 PM](#)
- #5 [Mar 9, 2010 12:09:36 PM](#)
- #3 [Mar 9, 2010 11:06:32 AM](#)
- #2 [Mar 9, 2010 10:47:59 AM](#)
- #1 [Mar 9, 2010 10:38:19 AM](#)

[for all](#) [for failures](#)

## Performance Trend

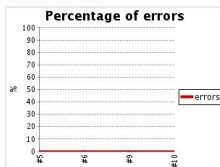
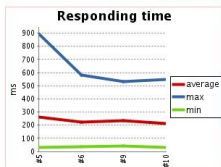
[Last Report](#)  
[Filter trend data](#)

### Test file: myTests1.jtl



[Trend report](#)

### Test file: myTests2.jtl



# Questions

---

- ▶ Common questions:
  - ▶ Can we survive christmas?

# Questions

---

- ▶ Common questions:
  - ▶ Can we survive christmas?
  - ▶ What is the maximum we can accomplish with the current setup?

# Questions

---

- ▶ **Common questions:**
  - ▶ Can we survive christmas?
  - ▶ What is the maximum we can accomplish with the current setup?
  - ▶ Does (APC|XCache|MemCache) really help us?

# Getting realistic settings

---

- ▶ Ask for real access logs:
  - ▶ Extract exact request model from those

# Getting realistic settings

---

- ▶ Ask for real access logs:
  - ▶ Extract exact request model from those
- ▶ Your customer usually only knows very broad values, like:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month



# Getting realistic settings

---

- ▶ Ask for real access logs:
  - ▶ Extract exact request model from those
- ▶ Your customer usually only knows very broad values, like:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Very seldom, that you get more:

# Getting realistic settings

---

- ▶ Ask for real access logs:
  - ▶ Extract exact request model from those
- ▶ Your customer usually only knows very broad values, like:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Very seldom, that you get more:
  - ▶ Ask for access statistics before Christmas and the ratio compared with regular months

# Getting realistic settings

---

- ▶ Ask for real access logs:
  - ▶ Extract exact request model from those
- ▶ Your customer usually only knows very broad values, like:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Very seldom, that you get more:
  - ▶ Ask for access statistics before Christmas and the ratio compared with regular months
  - ▶ Ask for hours with the highest conversion rates

# Getting realistic settings

---

- ▶ Ask for real access logs:
  - ▶ Extract exact request model from those
- ▶ Your customer usually only knows very broad values, like:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Very seldom, that you get more:
  - ▶ Ask for access statistics before Christmas and the ratio compared with regular months
  - ▶ Ask for hours with the highest conversion rates
  - ▶ Maybe get the aggregated access logs from existing similar shops

# Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month

# Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Per day:  $1.000.000/26 = 38.500$  (non-business)

# Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Per day:  $1.000.000/26 = 38.500$  (non-business)
- ▶ Per hour:  $38.500/12 = 3.200$  (national shop)

# Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Per day:  $1.000.000/26 = 38.500$  (non-business)
- ▶ Per hour:  $38.500/12 = 3.200$  (national shop)
- ▶ Peak hour:  $3.200 * 8 = 25.500$  (18:00 to 19:00)



## Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Per day:  $1.000.000/26 = 38.500$  (non-business)
- ▶ Per hour:  $38.500/12 = 3.200$  (national shop)
- ▶ Peak hour:  $3.200 * 8 = 25.500$  (18:00 to 19:00)
- ▶ Per second:  $25.500/3600 = 7PI/s$

## Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Per day:  $1.000.000/26 = 38.500$  (non-business)
- ▶ Per hour:  $38.500/12 = 3.200$  (national shop)
- ▶ Peak hour:  $3.200 * 8 = 25.500$  (18:00 to 19:00)
- ▶ Per second:  $25.500/3600 = 7PI/s$
- ▶ Add Christmas / Easter bonus

## Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Per day:  $1.000.000/26 = 38.500$  (non-business)
- ▶ Per hour:  $38.500/12 = 3.200$  (national shop)
- ▶ Peak hour:  $3.200 * 8 = 25.500$  (18:00 to 19:00)
- ▶ Per second:  $25.500/3600 = 7PI/s$
- ▶ Add Christmas / Easter bonus
- ▶ Add launch bonus

# Example calculation

---

- ▶ Customer provided values, for a classic webshop:
  - ▶ 1.000.000 PIs per month
  - ▶ 30.000 sold articles per month
  - ▶ 45.000 registrations per month
- ▶ Per day:  $1.000.000/26 = 38.500$  (non-business)
- ▶ Per hour:  $38.500/12 = 3.200$  (national shop)
- ▶ Peak hour:  $3.200 * 8 = 25.500$  (18:00 to 19:00)
- ▶ Per second:  $25.500/3600 = 7PI/s$
- ▶ Add Christmas / Easter bonus
- ▶ Add launch bonus
- ▶ So ... **50 PI/s** should be safe?
  - ▶ Spare resources for scaling are always a business decision
  - ▶ Provide with trade-off: Costs vs. downtime / slowness
  - ▶ Fail gracefully

# Verification

---

- ▶ Do your requests actually model customer requirements?
  - ▶ Compare generated access logs with real access logs
  - ▶ Compare user registrations / checkouts per hour with requested values

# Outline

---

Motivation

Conclusion

# Conclusion

---

- ▶ Plan your test scenario

# Conclusion

---

- ▶ Plan your test scenario
- ▶ Use realistic thresholds



# Conclusion

---

- ▶ Plan your test scenario
- ▶ Use realistic thresholds
- ▶ Choose the right tool

# Conclusion

---

- ▶ Plan your test scenario
- ▶ Use realistic thresholds
- ▶ Choose the right tool

with care

# Thanks for Listening

---

Rate this talk: <https://joind.in/7842>

# Thanks for Listening

---

Rate this talk: <https://joind.in/7842>

## Stay in touch

- ▶ Kore Nordmann
- ▶ [kore@qafoo.com](mailto:kore@qafoo.com)
- ▶ [@koredn](#) / [@qafoo](#)

Rent a web quality expert:  
<http://qafoo.com>