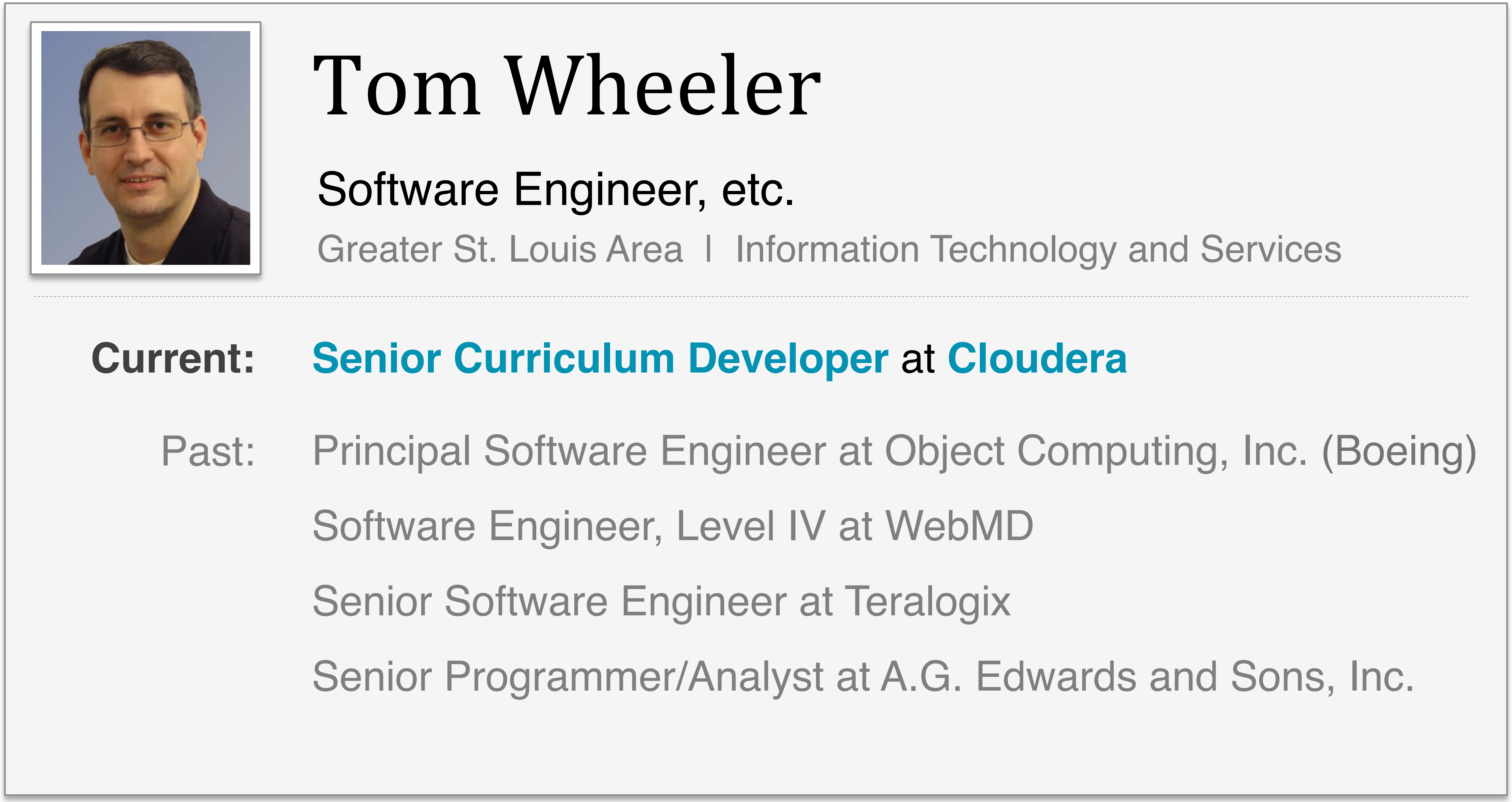
Testing Hadoop Applications

Tom Wheeler

About The Presenter…



Tom Wheeler

Software Engineer, etc.

Greater St. Louis Area | Information Technology and Services

Current: **Senior Curriculum Developer** at **Cloudera**

Past: Principal Software Engineer at Object Computing, Inc. (Boeing)

Software Engineer, Level IV at WebMD

Senior Software Engineer at Teralogix

Senior Programmer/Analyst at A.G. Edwards and Sons, Inc.

About The Presentation…

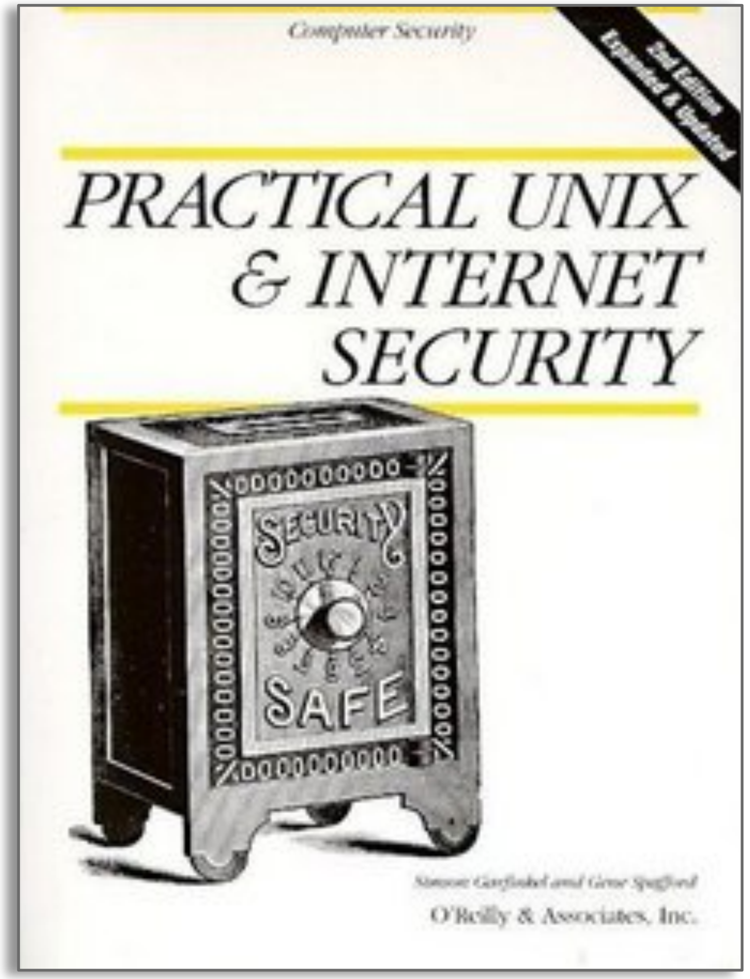
- What's ahead
  - Unit Testing
  - Integration Testing
  - Performance Testing
  - Diagnostics

About The Audience…

- What I expect from you
  - Basic knowledge of Apache Hadoop
  - Basic understanding of MapReduce (Java)
- But I'll assume no particular knowledge of testing

Fundamental Concepts

- What is testing?
  - In my view, it's an application of computer security

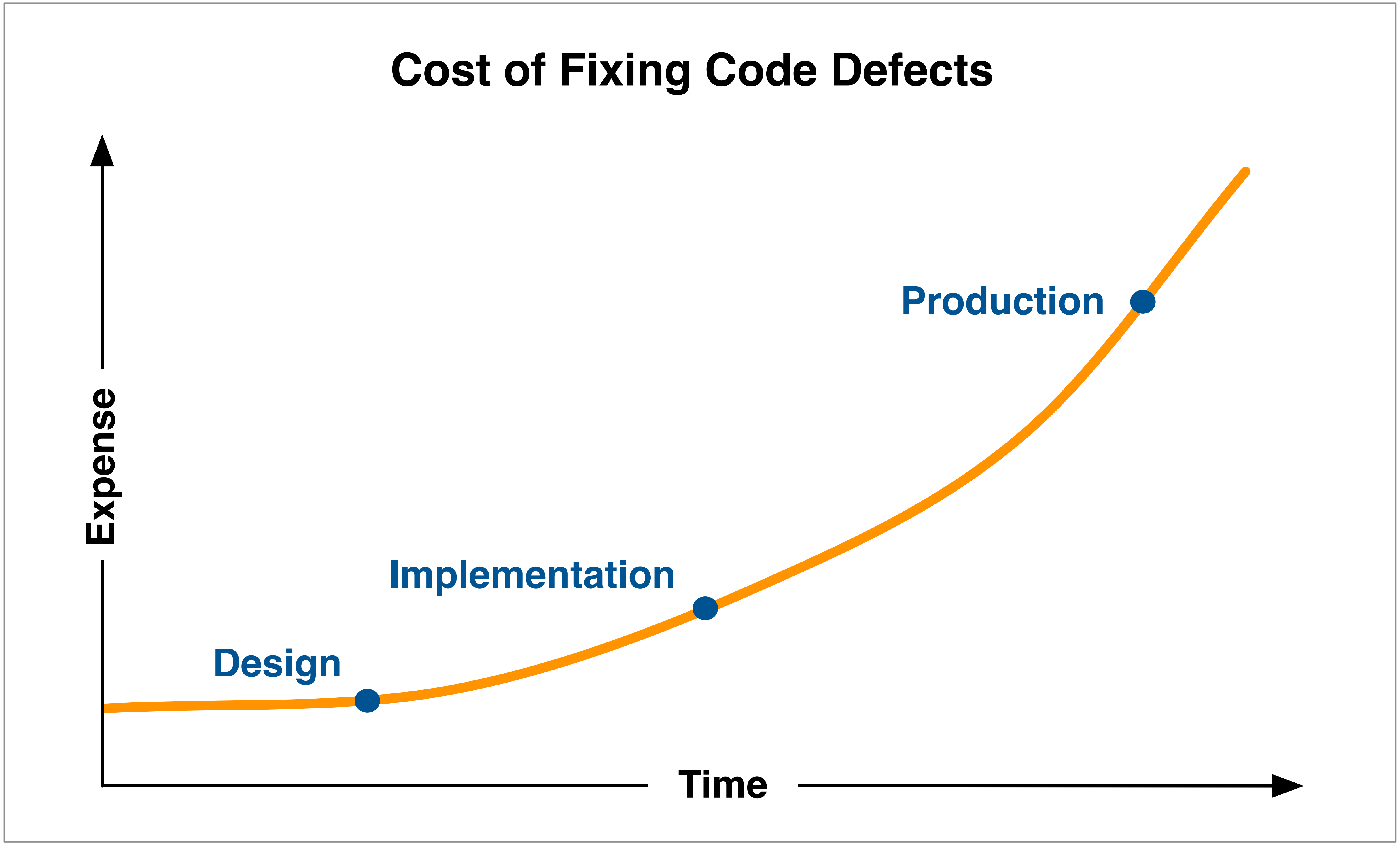

Computer Security
2nd Edition Expanded & Updated
PRACTICAL UNIX & INTERNET SECURITY
SECURITY
SAFE
Simson Garfinkel and Gene Spafford
O'Reilly & Associates, Inc.

> “A computer is secure if you can depend on it and its software to behave as you expect.”

Definitions

- Unit testing
    - Verifies function of each “unit” in isolation
- Integration testing
    - Verifies that the system works as a whole
- Performance testing
    - Verifies that code makes efficient use of resources
- Diagnostics
    - More about forensics than testing

Why is Testing Important?



Unit Testing

Overview of Unit Testing

- Verifies correctness for a unit of code
- Key features of unit tests
  - Simple
  - Isolated
  - Deterministic
  - Automated

Benefits of Unit Tests

- An investment in code quality
  - Can prevent regressions
- Actually saves development time
  - Tests run much faster than Hadoop jobs
  - Helps with refactoring

Introducing JUnit

- JUnit is a popular library for Java unit testing
  - Open source (CPL)
  - Add JAR file to your project
  - Inspired many “xUnit” libraries for other languages

JUnit Basics

Import statements removed for brevity

Our class

```java
public class Calculator {
   public int add(int first, int second) {
      return first + second;
   }
}
```

Our test

```java
public class CalculatorTest {

   private Calculator calc;

   @Before
   public void setUp() {
      calc = new Calculator();
   }

   @Test
   public void testAdd() {
      assertEquals(8, calc.add(5, 3));
   }
}
```

Log Event Counting Example

- Let’s look at a simple MapReduce example
- Our goal is to summarize log events by level
  - Mapper: Parses log to extract level
  - Reducer: Sums up occurrences by level
- Seeing the data flow first will help illustrate the job

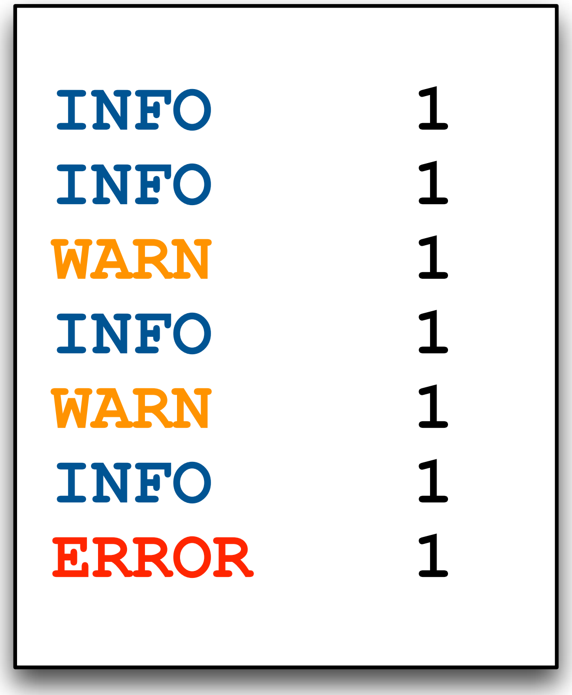
Mapper Input

- Log data produced by an application using Log4J

```
2012-09-06 22:16:49.391 CDT INFO "This can wait"
2012-09-06 22:16:49.392 CDT INFO "Blah blah"
2012-09-06 22:16:49.394 CDT WARN "Hmmm..."
2012-09-06 22:16:49.395 CDT INFO "More blather"
2012-09-06 22:16:49.397 CDT WARN "Hey there"
2012-09-06 22:16:49.398 CDT INFO "Spewing data"
2012-09-06 22:16:49.399 CDT ERROR "Oh boy!"
```

Mapper Output

- Key is log level parsed from a single line in the file
- Value is a literal `1` (since there's one level per line)



```
INFO      1
INFO      1
WARN      1
INFO      1
WARN      1
INFO      1
ERROR     1
```

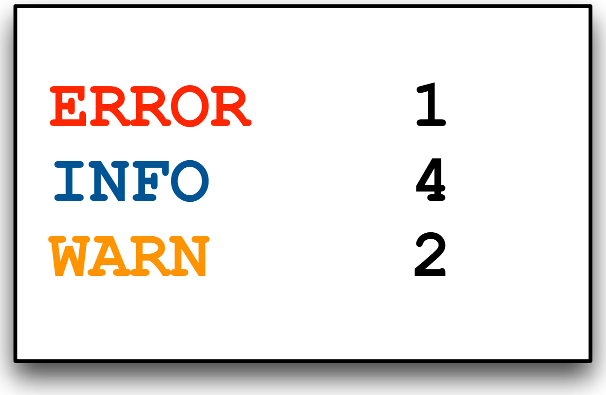
Reducer Input

- Hadoop sorts and groups the keys

```
ERROR     [1]
INFO      [1, 1, 1, 1]
WARN      [1, 1]
```

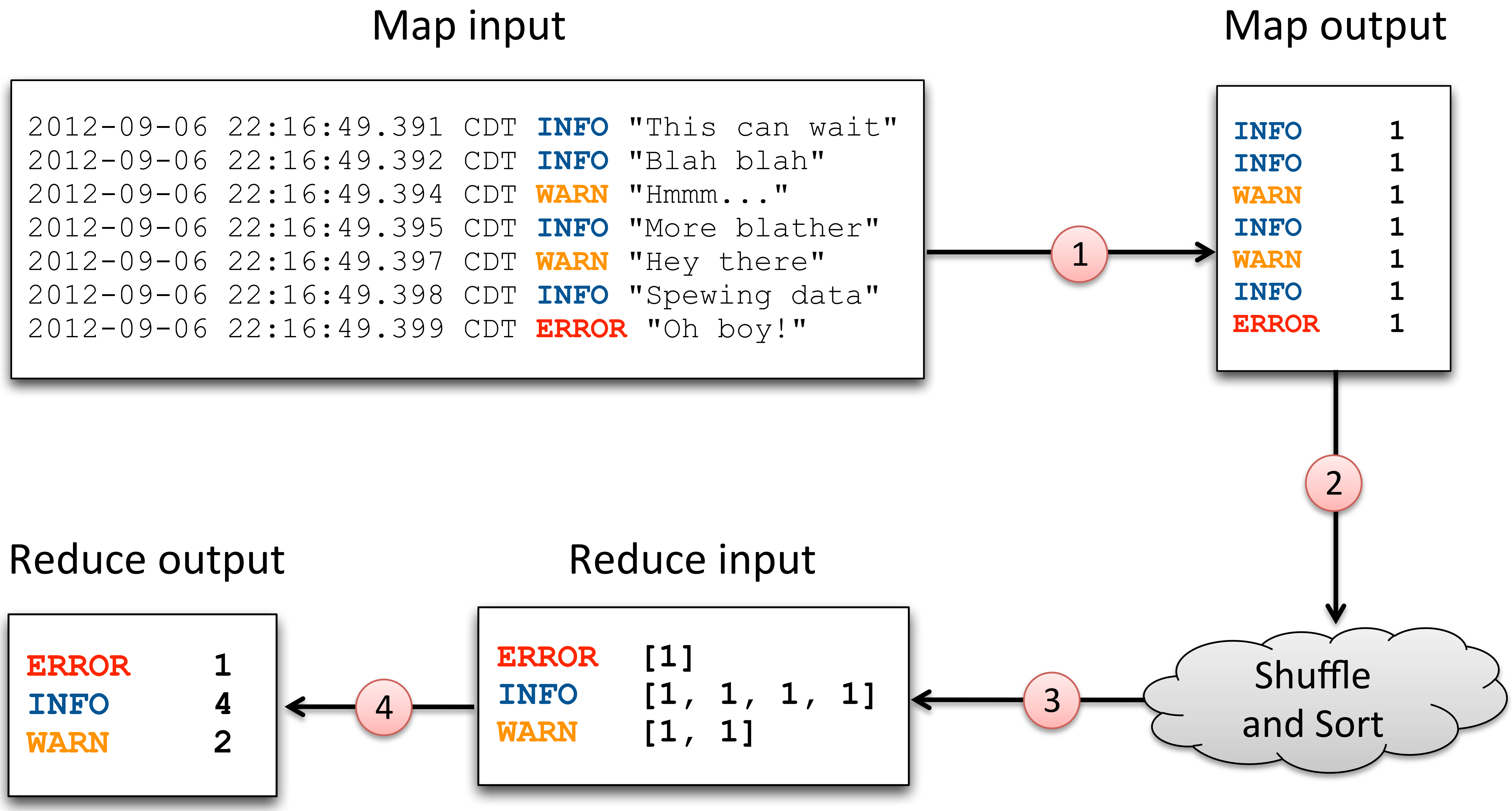
Reducer Output

- The final output is a per-level summary



```
ERROR        1
INFO         4
WARN         2
```

Data Flow for Entire MapReduce Job



Map input

```
2012-09-06 22:16:49.391 CDT INFO "This can wait"
2012-09-06 22:16:49.392 CDT INFO "Blah blah"
2012-09-06 22:16:49.394 CDT WARN "Hmmm..."
2012-09-06 22:16:49.395 CDT INFO "More blather"
2012-09-06 22:16:49.397 CDT WARN "Hey there"
2012-09-06 22:16:49.398 CDT INFO "Spewing data"
2012-09-06 22:16:49.399 CDT ERROR "Oh boy!"
```

Map output

```
INFO      1
INFO      1
WARN      1
INFO      1
WARN      1
INFO      1
ERROR     1
```

Reduce input

```
ERROR   [1]
INFO    [1, 1, 1, 1]
WARN    [1, 1]
```

Reduce output

```
ERROR     1
INFO      4
WARN      2
```

Show Me the Code…

- Let’s examine the code for this job
  - Then I’ll run it
  - And soon we’ll see how to test and improve it

What’s Wrong With This?

- Mapper is complex and hard to test
  - How could we improve it?
- Refactor to decouple business logic from Hadoop API

Unit Testing and External Dependencies

- We can validate core business logic with JUnit
  - But our Mapper and Reducer still aren't fully tested
  - How can we verify *them*?

Introducing MRUnit

- It’s a JUnit extension for testing MapReduce code
  - Open source (Apache licensed)
  - Active development team
- Simulates much of Hadoop’s core API, including
  - `InputSplit`
  - `OutputCollector`
  - `Reporter`
  - `Counters`

Apache MRUnit™

MRUnit Demo

- I'll now demonstrate how to use MRUnit
    - Testing the Mapper
    - Testing the Reducer

Limitations of MRUnit

- There are a few things MRUnit can’t test (yet)
  - Multiple lines of input for a Mapper
  - Jobs that use DistributedCache
  - Partitioners
  - Hadoop Streaming

Integration Testing

Overview of Integration Testing

- Verifies that the system works as a whole
- This can mean two things
  - Your units of code work with one another
  - Your code works with the underlying system

Testing Mappers and Reducers Together

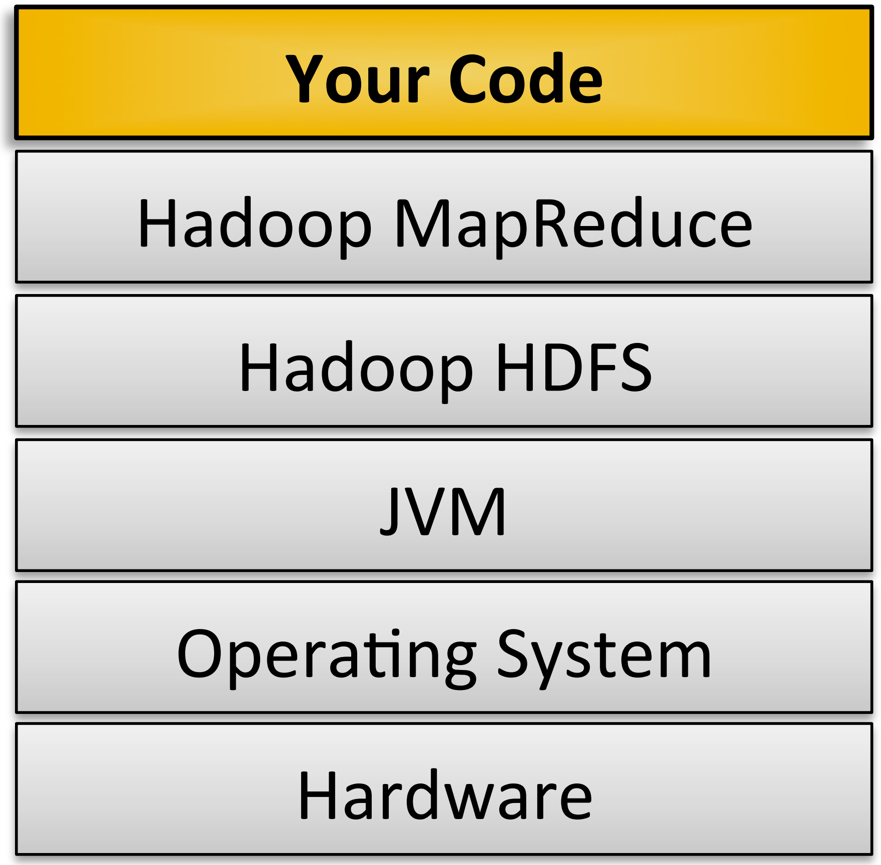
- Unit tests verify Mappers and Reducers separately
  - Also need to ensure they work *together*
- MRUnit can help here too

MiniMRCluster and MiniDFSCluster

- MRUnit can test Mapper and Reducer integration
  - But not the *entire job*
- Two “mini-cluster” classes
  - `MiniMRCluster` simulates MapReduce
  - `MiniDFSCluster` simulates HDFS
- Let’s see an example…

Integration Testing the Hadoop Stack

- Your code depends on many other components
  - Your code doesn’t function properly unless they do



Your Code

Hadoop MapReduce

Hadoop HDFS

JVM

Operating System

Hardware

Integration Testing with iTest

- iTest is a product of Apache Bigtop
  - Can be used for custom integration testing
- iTest is written in Groovy
  - The tests themselves can be in Java or Groovy

iTest Example

```
public class RunHadoopJobsTest {

   @Test
   public void testJobSuccess() {
     Shell sh = new Shell("/bin/bash -s");
     sh.exec("hadoop jar /tmp/myjob.jar MyDriver input output");
     assertEquals(0, sh.getRet());
   }

   @Test
   public void testJobFail() {
     Shell sh = new Shell("/bin/bash -s");
     sh.exec("hadoop jar /tmp/myjob.jar MyDriver bogus_args");
     assertEquals(1, sh.getRet());
   }
}
```

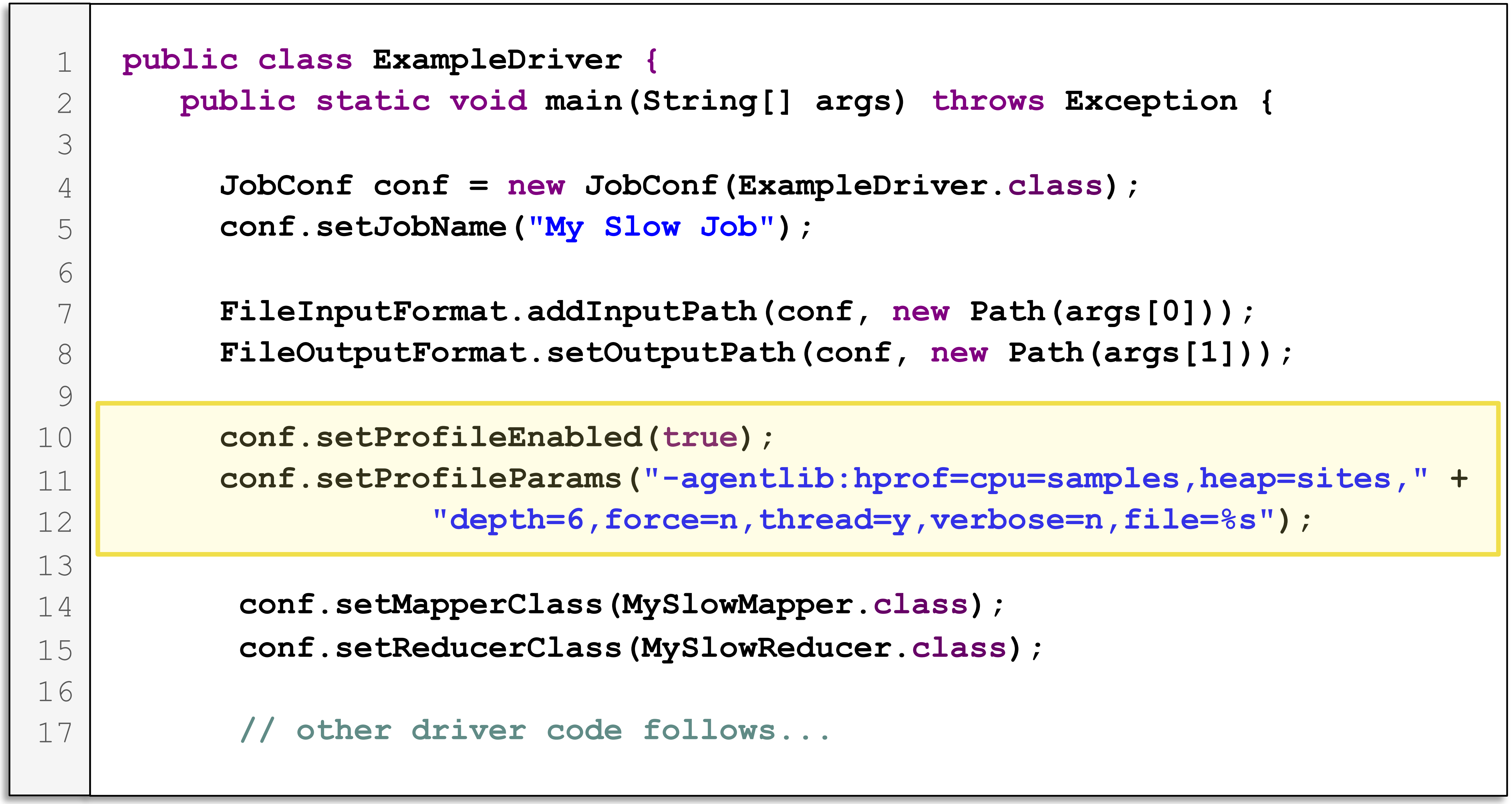
Import statements removed for brevity

Performance Testing

Resource Utilization

- Optimization is a tradeoff between resources
  - CPU
  - Memory
  - Disk I/O
  - Network I/O
  - Developer time
  - Hardware cost

Profiling



```java
public class ExampleDriver {
   public static void main(String[] args) throws Exception {

     JobConf conf = new JobConf(ExampleDriver.class);
     conf.setJobName("My Slow Job");

     FileInputFormat.addInputPath(conf, new Path(args[0]));
     FileOutputFormat.setOutputPath(conf, new Path(args[1]));

     conf.setProfileEnabled(true);
     conf.setProfileParams("-agentlib:hprof=cpu=samples,heap=sites," +
                "depth=6,force=n,thread=y,verbose=n,file=%s");

      conf.setMapperClass(MySlowMapper.class);
      conf.setReducerClass(MySlowReducer.class);

      // other driver code follows...
```

Profiling

- One `.profile` file per Map / Reduce task attempt

```
$ ls *.profile
attempt_201206281229_0070_m_000000_0.profile
attempt_201206281229_0070_r_000000_0.profile
attempt_201206281229_0070_m_000001_0.profile
attempt_201206281229_0070_r_000001_0.profile
attempt_201206281229_0070_m_000002_0.profile
attempt_201206281229_0070_r_000002_0.profile
```

Diagnostics

Overview of Diagnostics

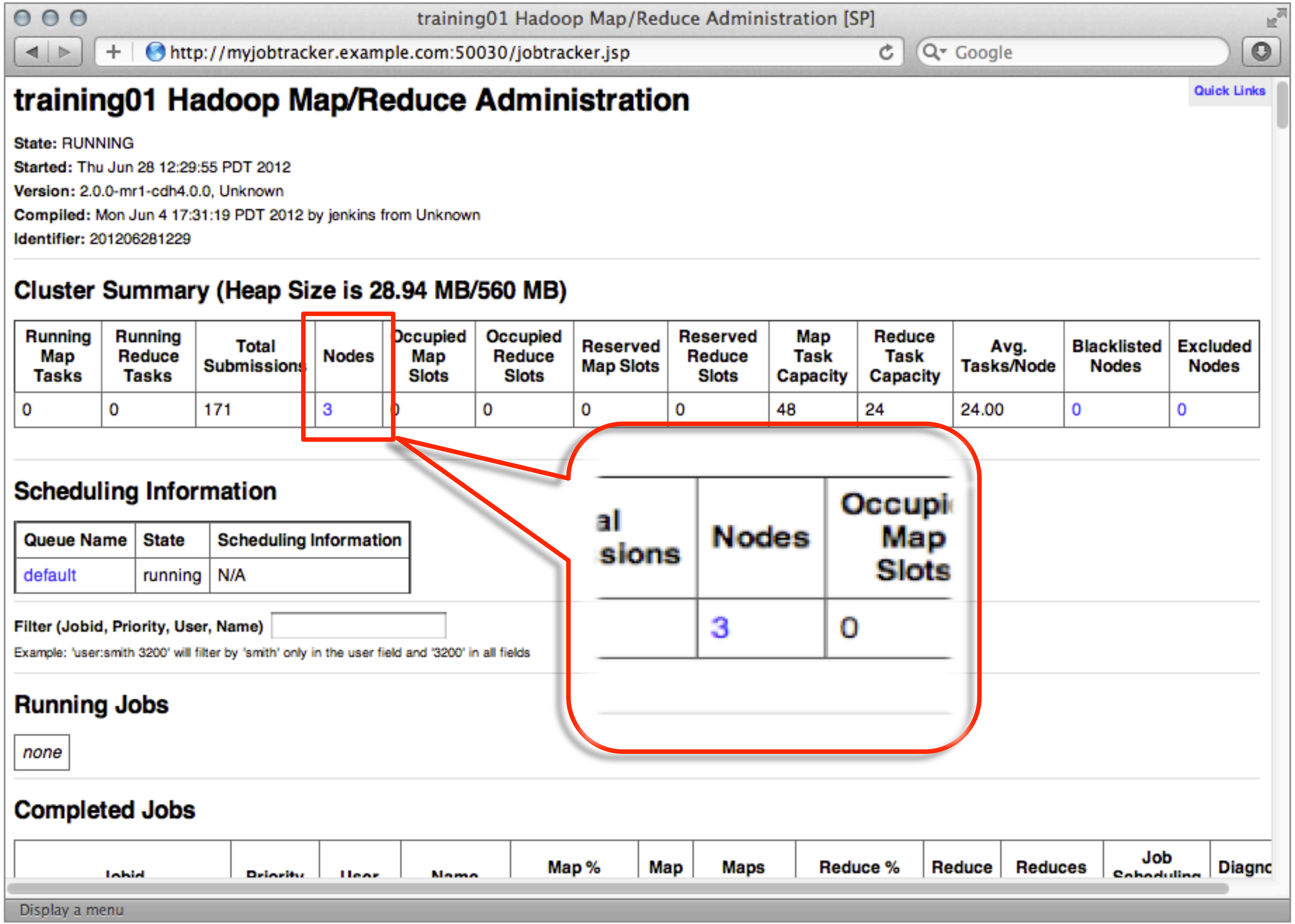
- Testing is proactive
  - Diagnostics are reactive
- Many diagnostic tools available, including
  - Web UI
  - Logs
  - Counters
  - Job history

Web UI

- Each Hadoop daemon has its own Web application

Daemon	URL
JobTracker	**`http://hostname:50030/`**
TaskTracker	`http://hostname:50060/`
NameNode	**`http://hostname:50070/`**
Secondary NameNode	`http://hostname:50090/`
DataNode	`http://hostname:50075/`

JobTracker Web UI Example


training01 Hadoop Map/Reduce Administration [SP]
http://myjobtracker.example.com:50030/jobtracker.jsp
Google
Quick Links
al
sions
Nodes
Occupi
Map
Slots
3
0
Display a menu

training01 Hadoop Map/Reduce Administration

State: RUNNING
Started: Thu Jun 28 12:29:55 PDT 2012
Version: 2.0.0-mr1-cdh4.0.0, Unknown
Compiled: Mon Jun 4 17:31:19 PDT 2012 by jenkins from Unknown
Identifier: 201206281229

Cluster Summary (Heap Size is 28.94 MB/560 MB)

Running Map Tasks	Running Reduce Tasks	Total Submissions	Nodes	Occupied Map Slots	Occupied Reduce Slots	Reserved Map Slots	Reserved Reduce Slots	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node	Blacklisted Nodes	Excluded Nodes
0	0	171	3	0	0	0	0	48	24	24.00	0	0

Scheduling Information

Queue Name	State	Scheduling Information
default	running	N/A

Filter (Jobid, Priority, User, Name)

Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

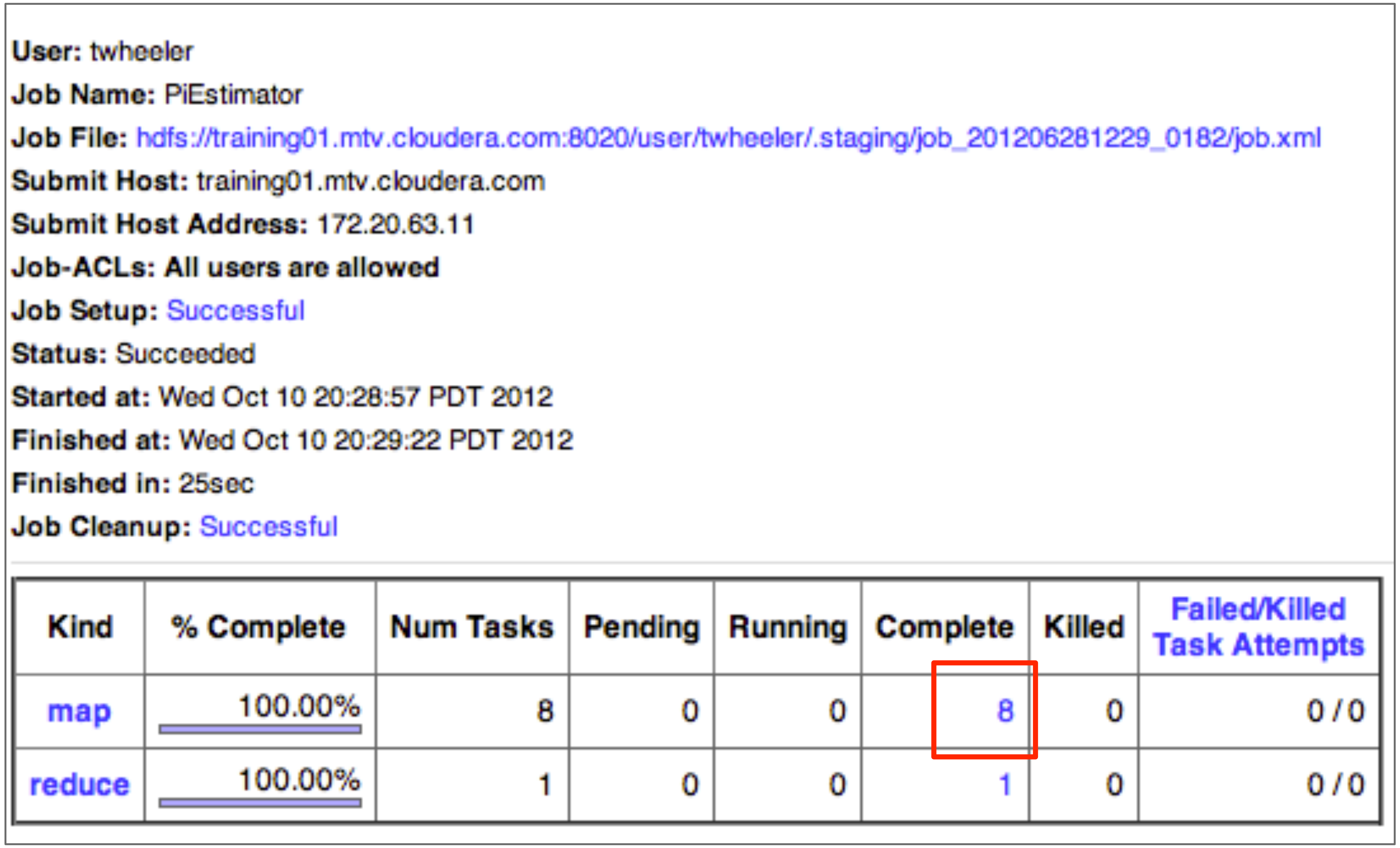
Running Jobs

none

Completed Jobs

Jobid	Priority	User	Name	Map %	Map	Maps	Reduce %	Reduce	Reduces	Job Scheduling	Diagnc

JobTracker Job History Page



User: twheeler

Job Name: PiEstimator

Job File: hdfs://training01.mtv.cloudera.com:8020/user/twheeler/.staging/job_201206281229_0182/job.xml

Submit Host: training01.mtv.cloudera.com

Submit Host Address: 172.20.63.11

Job-ACLs: All users are allowed

Job Setup: Successful

Status: Succeeded

Started at: Wed Oct 10 20:28:57 PDT 2012

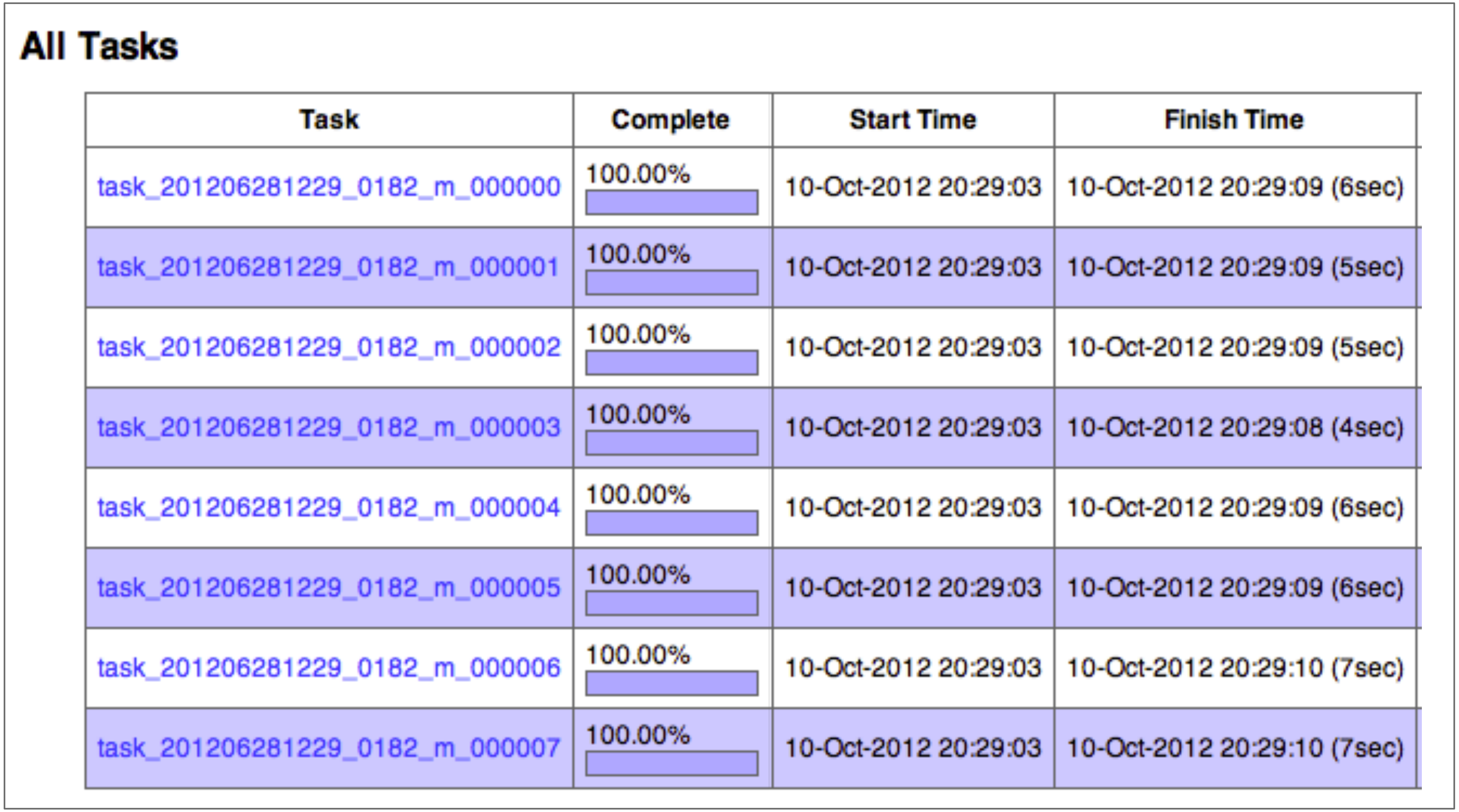
Finished at: Wed Oct 10 20:29:22 PDT 2012

Finished in: 25sec

Job Cleanup: Successful

Kind	% Complete	Num Tasks	Pending	Running	Complete	Killed	Failed/Killed Task Attempts
map	100.00%	8	0	0	8	0	0 / 0
reduce	100.00%	1	0	0	1	0	0 / 0

Task Attempt List



All Tasks

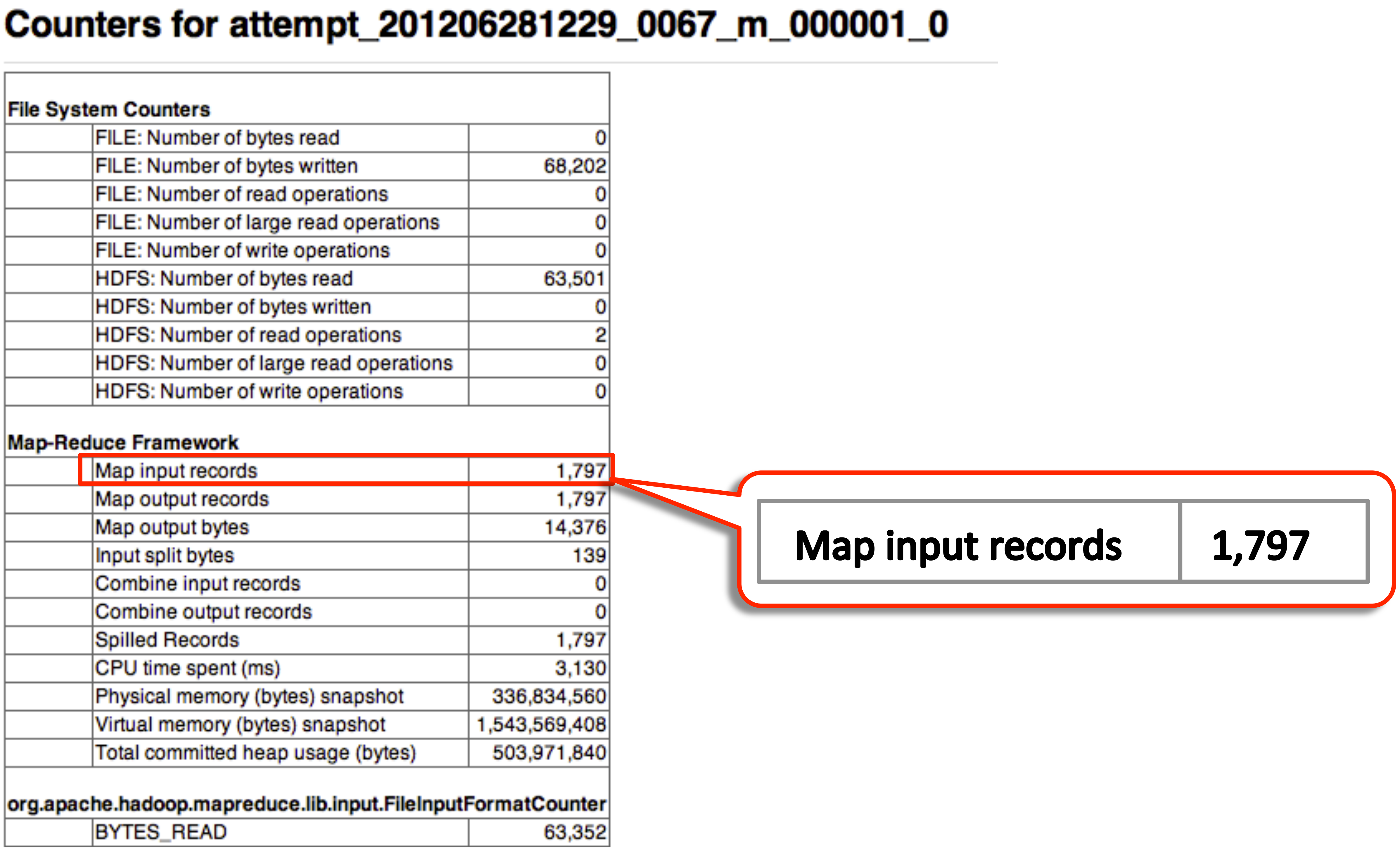
Task	Complete	Start Time	Finish Time
task_201206281229_0182_m_000000	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:09 (6sec)
task_201206281229_0182_m_000001	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:09 (5sec)
task_201206281229_0182_m_000002	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:09 (5sec)
task_201206281229_0182_m_000003	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:08 (4sec)
task_201206281229_0182_m_000004	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:09 (6sec)
task_201206281229_0182_m_000005	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:09 (6sec)
task_201206281229_0182_m_000006	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:10 (7sec)
task_201206281229_0182_m_000007	100.00%	10-Oct-2012 20:29:03	10-Oct-2012 20:29:10 (7sec)

Some columns removed to better fit the screen

Logs

- Most important logs for MapReduce developers
  - stdout
  - stderr
  - syslog
- Demo: How to debug error using Web UI and logs

Counters

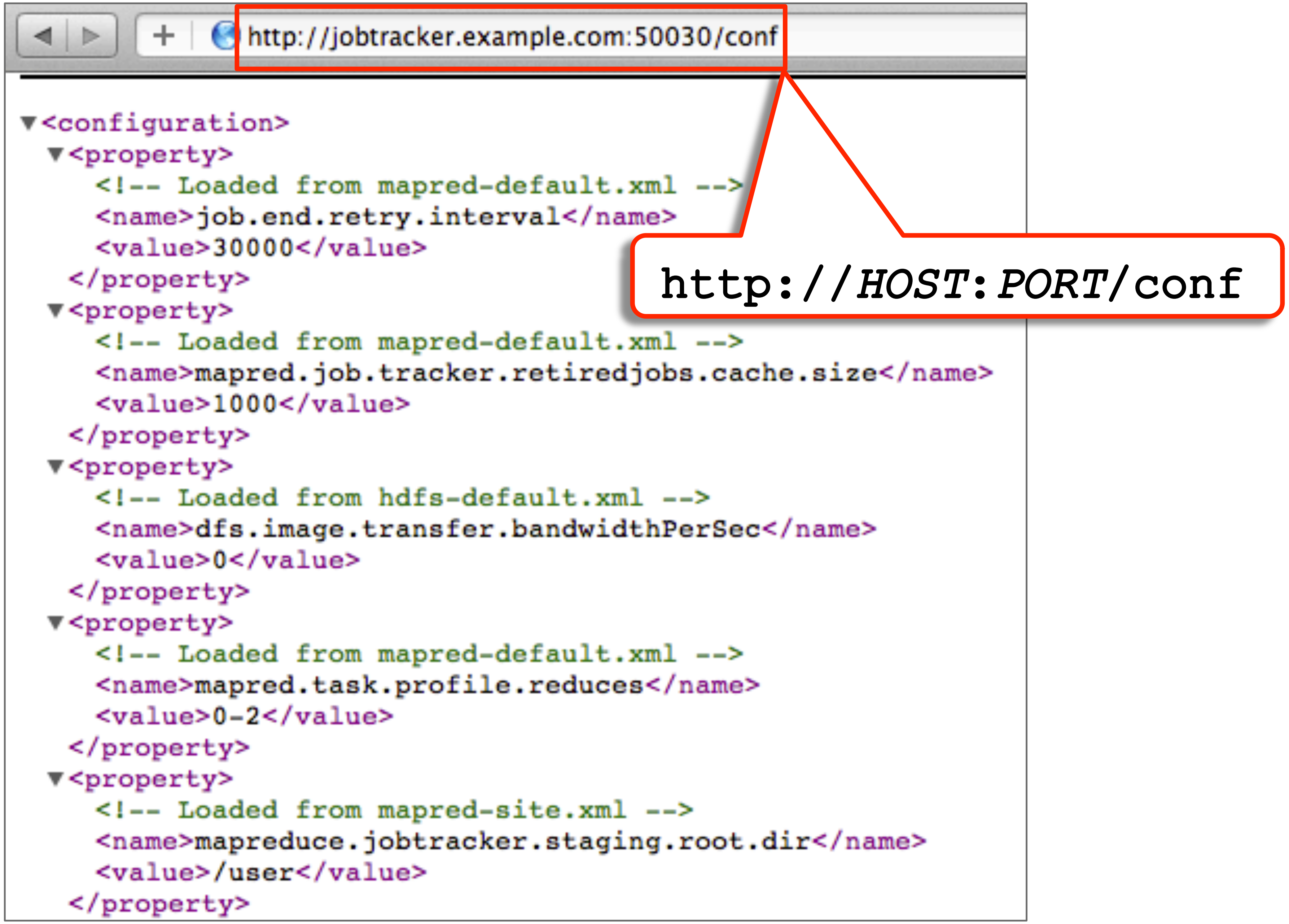


Counters for attempt_201206281229_0067_m_000001_0

File System Counters		
	FILE: Number of bytes read	0
	FILE: Number of bytes written	68,202
	FILE: Number of read operations	0
	FILE: Number of large read operations	0
	FILE: Number of write operations	0
	HDFS: Number of bytes read	63,501
	HDFS: Number of bytes written	0
	HDFS: Number of read operations	2
	HDFS: Number of large read operations	0
	HDFS: Number of write operations	0
Map-Reduce Framework		
	Map input records	1,797
	Map output records	1,797
	Map output bytes	14,376
	Input split bytes	139
	Combine input records	0
	Combine output records	0
	Spilled Records	1,797
	CPU time spent (ms)	3,130
	Physical memory (bytes) snapshot	336,834,560
	Virtual memory (bytes) snapshot	1,543,569,408
	Total committed heap usage (bytes)	503,971,840
org.apache.hadoop.mapreduce.lib.input.FileInputFormatCounter		
	BYTES_READ	63,352

Map input records	**1,797**

Viewing Current Configuration

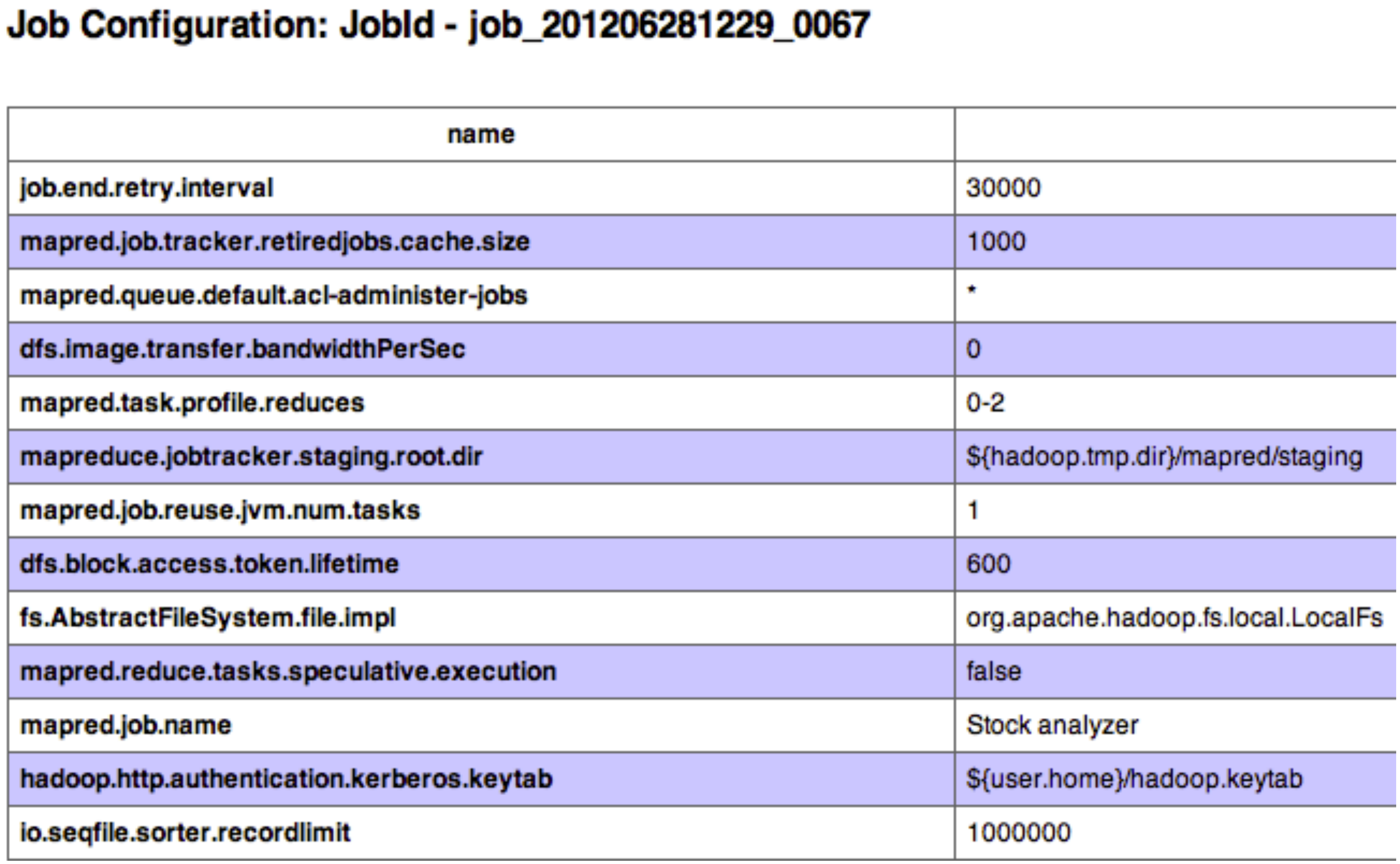


http://jobtracker.example.com:50030/conf

```
▼<configuration>
  ▼<property>
     <!-- Loaded from mapred-default.xml -->
     <name>job.end.retry.interval</name>
     <value>30000</value>
   </property>
  ▼<property>
     <!-- Loaded from mapred-default.xml -->
     <name>mapred.job.tracker.retiredjobs.cache.size</name>
     <value>1000</value>
   </property>
  ▼<property>
     <!-- Loaded from hdfs-default.xml -->
     <name>dfs.image.transfer.bandwidthPerSec</name>
     <value>0</value>
   </property>
  ▼<property>
     <!-- Loaded from mapred-default.xml -->
     <name>mapred.task.profile.reduces</name>
     <value>0-2</value>
   </property>
  ▼<property>
     <!-- Loaded from mapred-site.xml -->
     <name>mapreduce.jobtracker.staging.root.dir</name>
     <value>/user</value>
   </property>
```

`http://`*`HOST`*`:`*`PORT`*`/conf`

Job Properties



Job Configuration: JobId - job_201206281229_0067

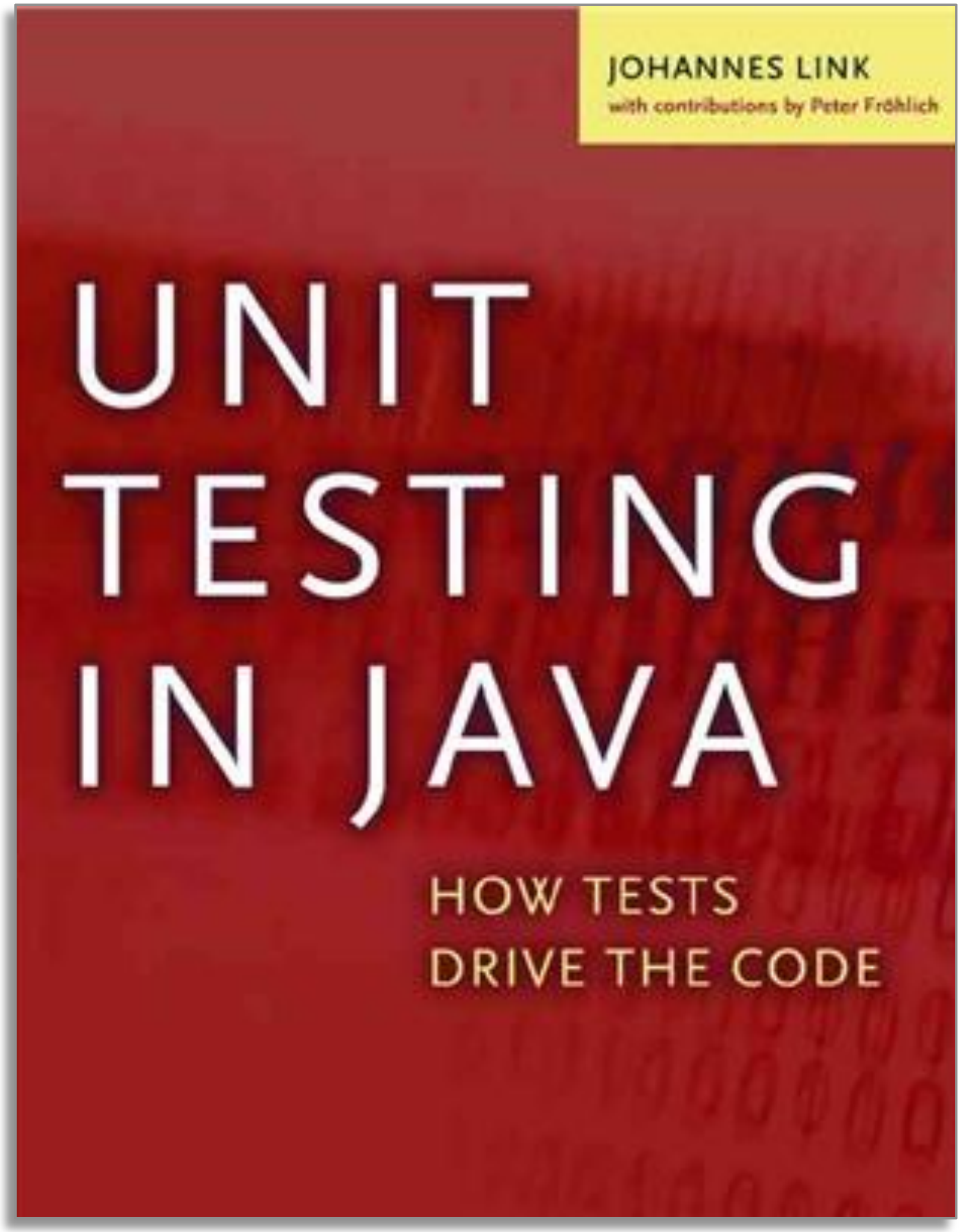
name	
job.end.retry.interval	30000
mapred.job.tracker.retiredjobs.cache.size	1000
mapred.queue.default.acl-administer-jobs	*
dfs.image.transfer.bandwidthPerSec	0
mapred.task.profile.reduces	0-2
mapreduce.jobtracker.staging.root.dir	${hadoop.tmp.dir}/mapred/staging
mapred.job.reuse.jvm.num.tasks	1
dfs.block.access.token.lifetime	600
fs.AbstractFileSystem.file.impl	org.apache.hadoop.fs.local.LocalFs
mapred.reduce.tasks.speculative.execution	false
mapred.job.name	Stock analyzer
hadoop.http.authentication.kerberos.keytab	${user.home}/hadoop.keytab
io.seqfile.sorter.recordlimit	1000000

Recap

- There are many types of testing
  - Unit
  - Integration
  - Performance
- Diagnostics help us find the problems we missed

Resources for More Info

- Johannes Link
- ISBN: 1-55860-868-0



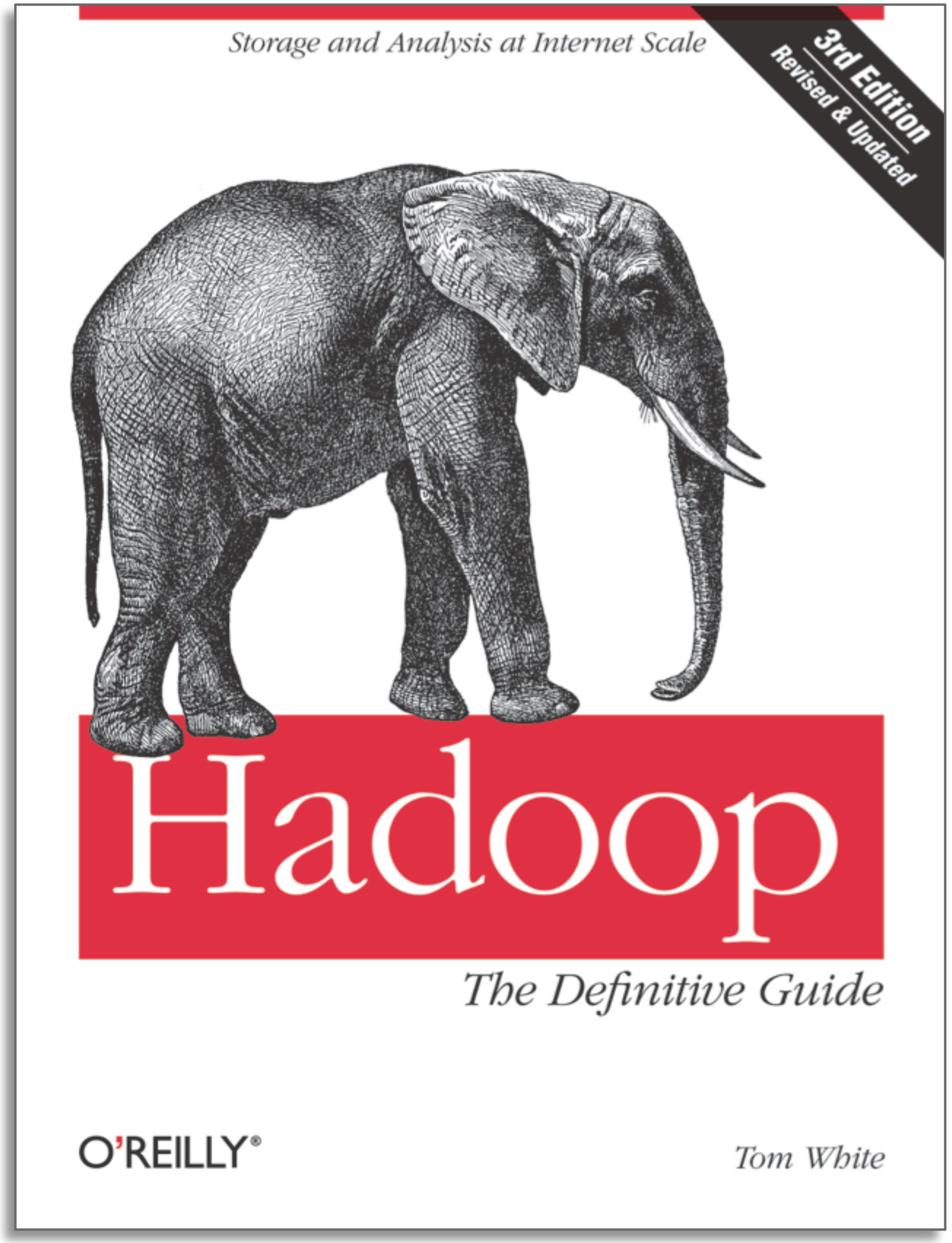
JOHANNES LINK
with contributions by Peter Fröhlich

UNIT TESTING IN JAVA

HOW TESTS DRIVE THE CODE

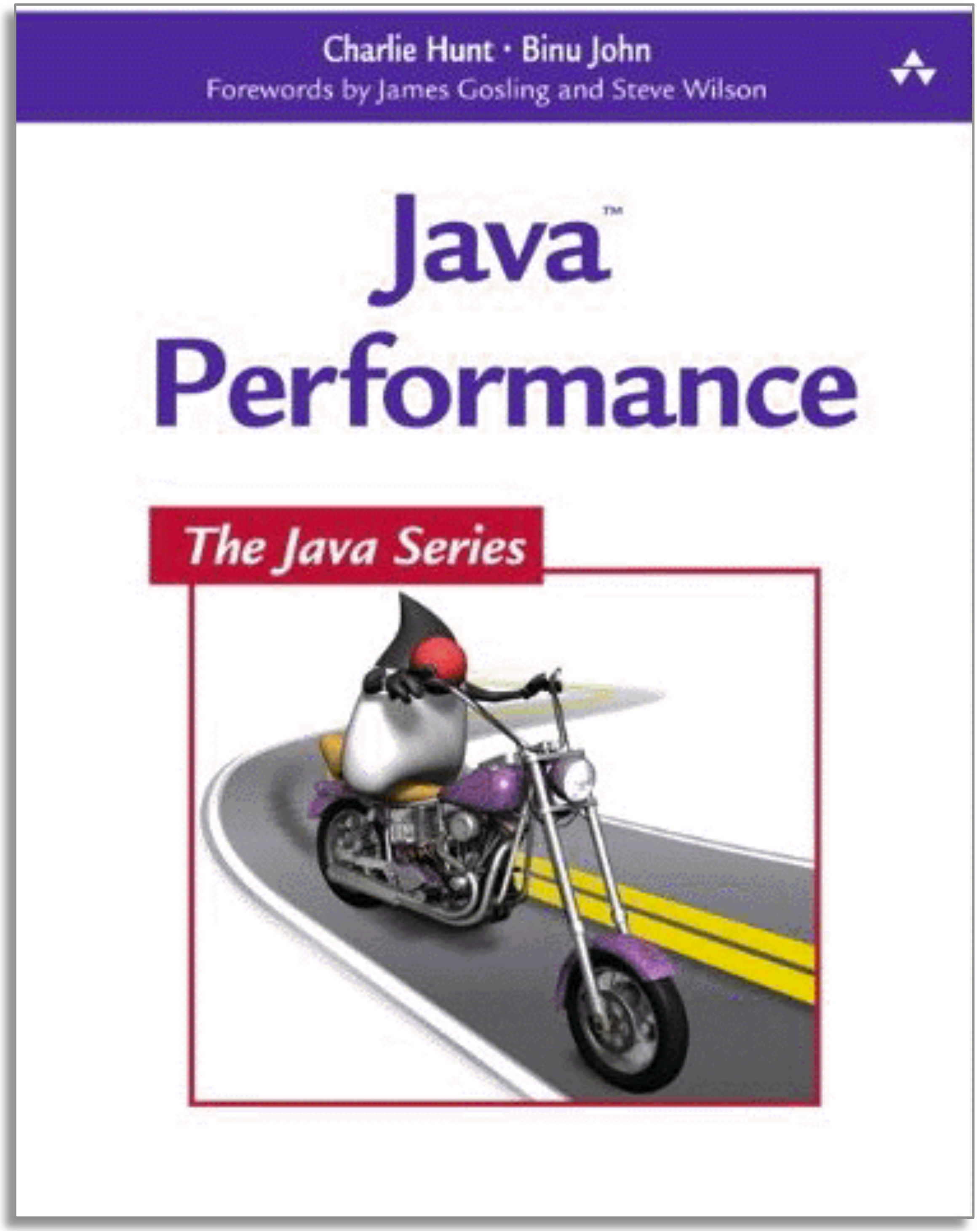
Resources for More Info (cont'd)

- Tom White
- ISBN: 1-449-31152-0



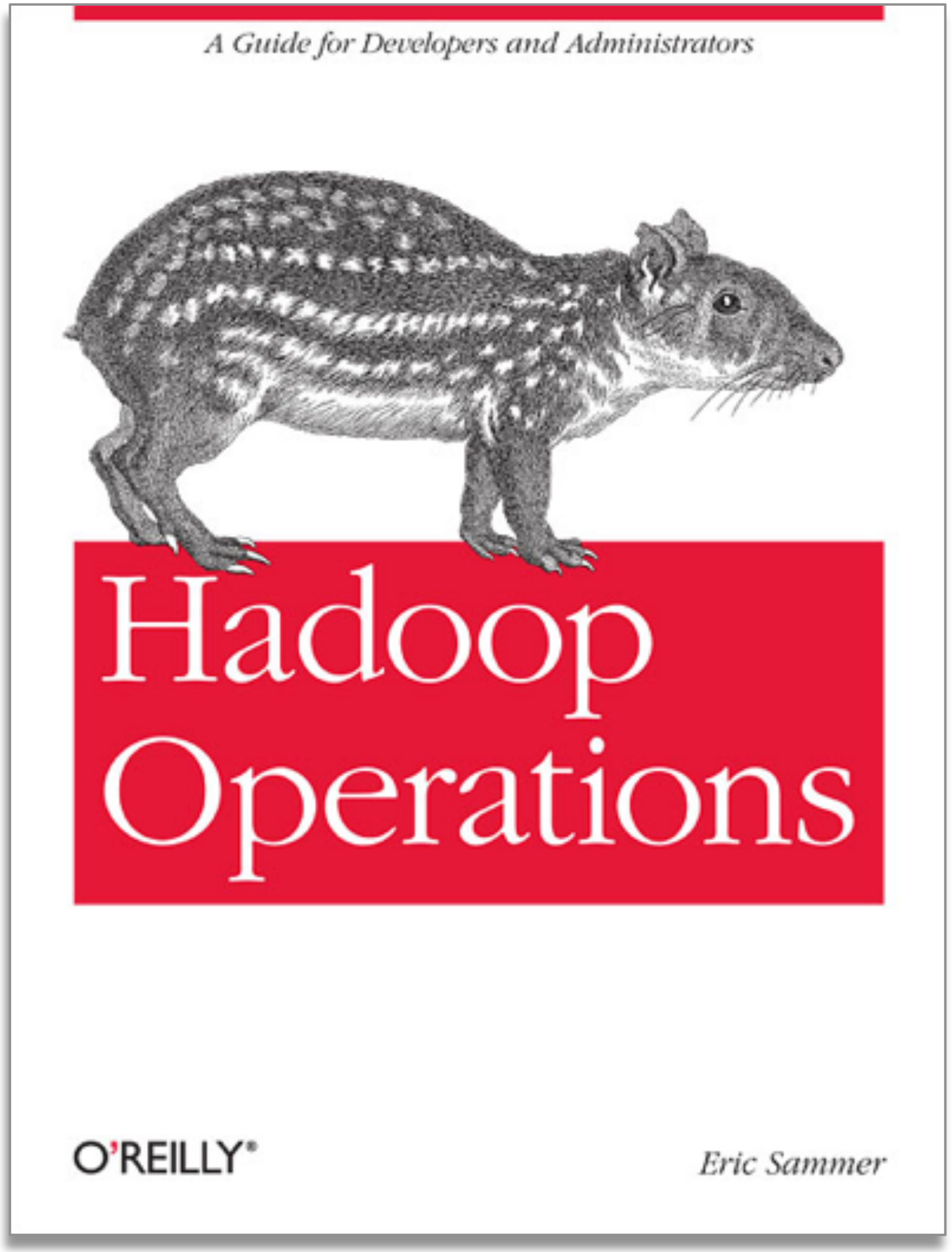
Resources for More Info (cont'd)

- Charlie Hunt and Binu John
- ISBN: 0-137-14252-8



Resources for More Info (cont'd)

- Eric Sammer
- ISBN: 1-449-32705-2



Conclusion

- Testing helps develop trust in a system
  - It should behave as you expect it to

cloudera®
Ask Bigger Questions

Thank You!

Tom Wheeler, Senior Curriculum Developer, Cloudera