Introduction to SOEN 691: Mining Large Software System Data for DevOps

Weiyi Shang

UNIVERSITÉ
Concordia
UNIVERSITY

Who is this guy?

Academia

Queen's University

M.Sc., Ph.D., Post-Doc
Sept. 2008-
July. 2015

Industry

BlackBerry

Performance Engineer
Sept. 2010-
Aug. 2014

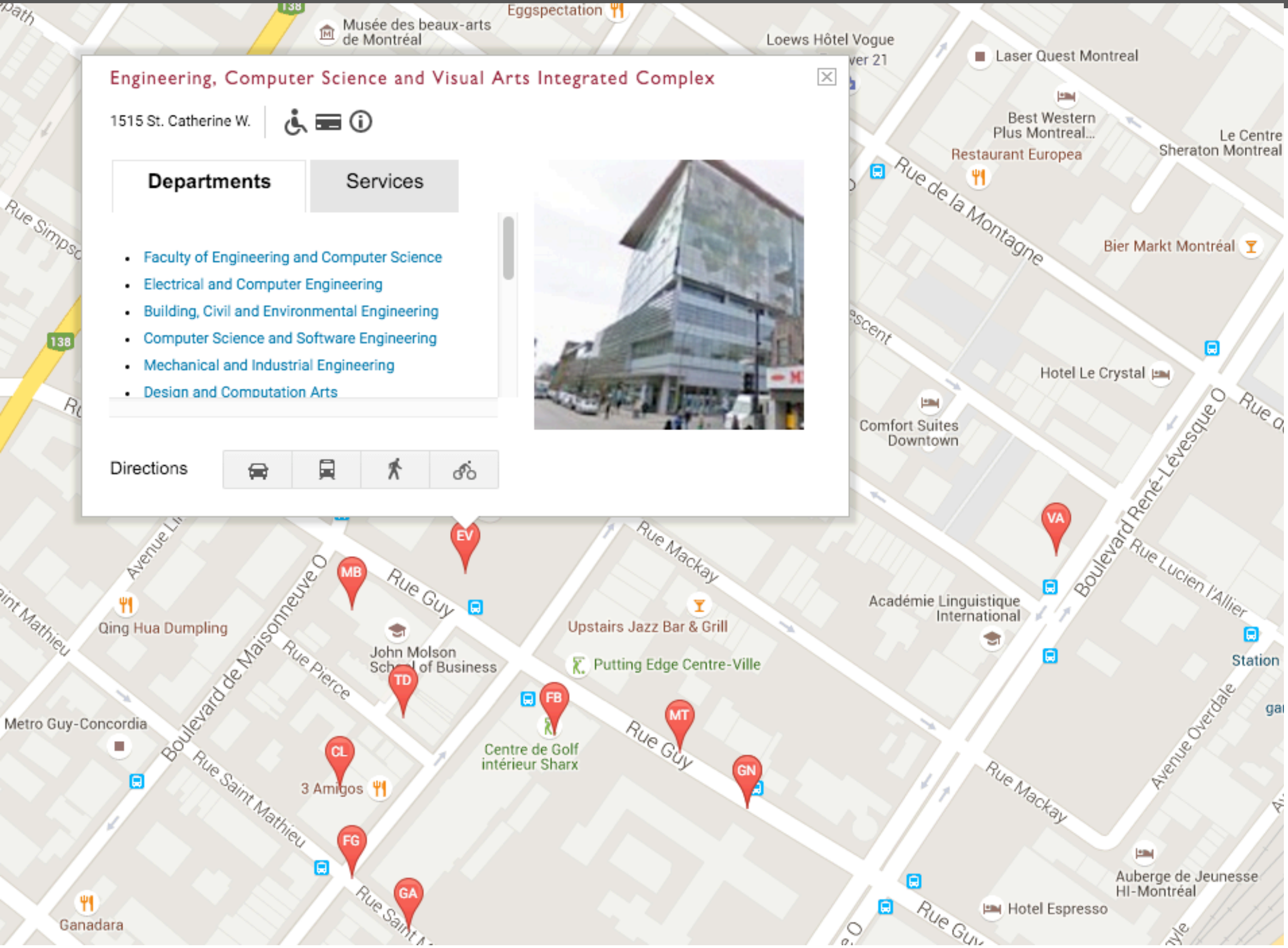
Prof. Shang’s research

Mining large software data

Software engineering for large systems

Where am I?

Where am I?



Engineering, Computer Science and Visual Arts Integrated Complex

1515 St. Catherine W.

Departments | Services

- Faculty of Engineering and Computer Science
- Electrical and Computer Engineering
- Building, Civil and Environmental Engineering
- Computer Science and Software Engineering
- Mechanical and Industrial Engineering
- Design and Computation Arts

Directions

Room # 3.129

Email: shang@encs.concordia.ca

What are we doing here?

What are we doing here?

Learning about the how to leverage the large scale data of software systems in order to assist in DevOps. Topics include

(1) Logging
(2) Software performance
(3) Large-scale testing
(4) Empirical studies on software data
(5) Software configuration

Time of the class VERY IMPORTANT

1:30 to 4:00 PM
I will try to be here 15 minutes before class for Q&A
I can't do Q&A after the class

What if I want to meet with you?

Need advise:
Send me an email, I will arrange a meeting in person.

Technical or course logistic questions:
POD/TA of the course:
Mehran Hassani:
mehran.hassany@gmail.com

What do I need to survive?

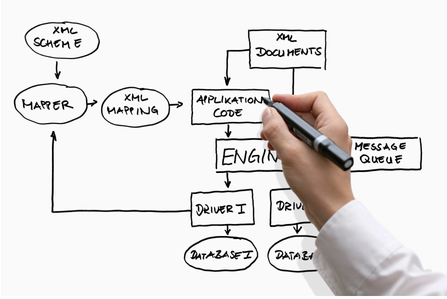
What do I need to survive?

This is NOT a lecture course!

Good discussion, expressing your opinion.
Read papers.
A good project.

What is DevOps?

Software Devlopment

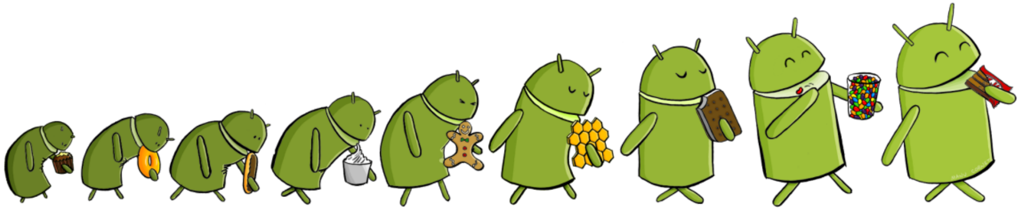


Design and specification

Coding

Testing

Release engineering

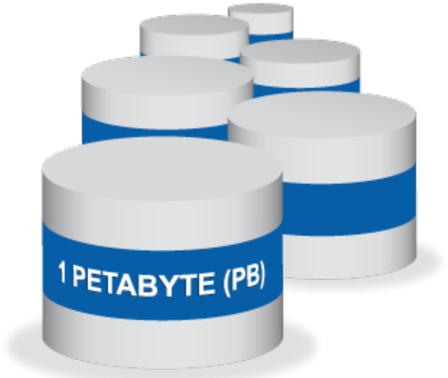


Evolution

Software Operation

Monitoring

Troubleshooting



Capacity planning

Anomaly detection

Q&A

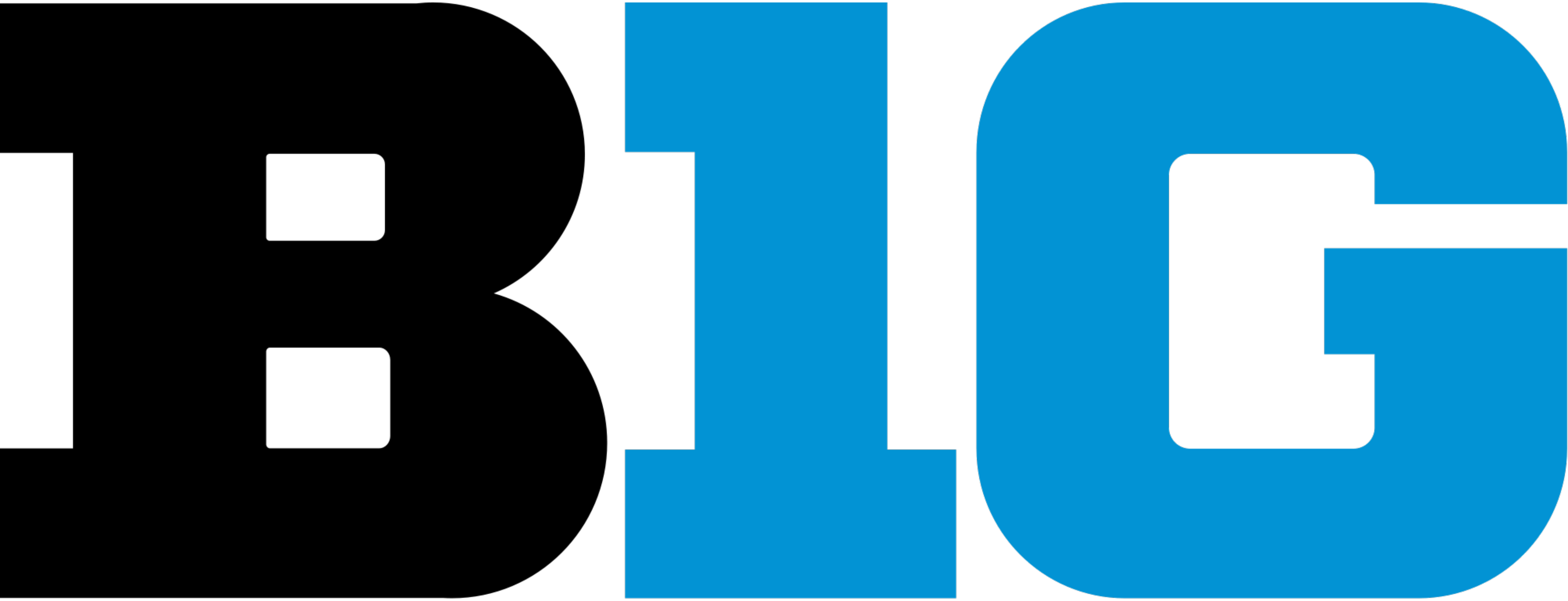
Configuration Tuning

What is DevOps?

DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.

DevOps is also characterized by operations staff making use many of the same techniques as developers for their systems work.

Context of DevOps



Ultra-large-scale Systems (ULSS) : Millions of Users, Billions of Transactions

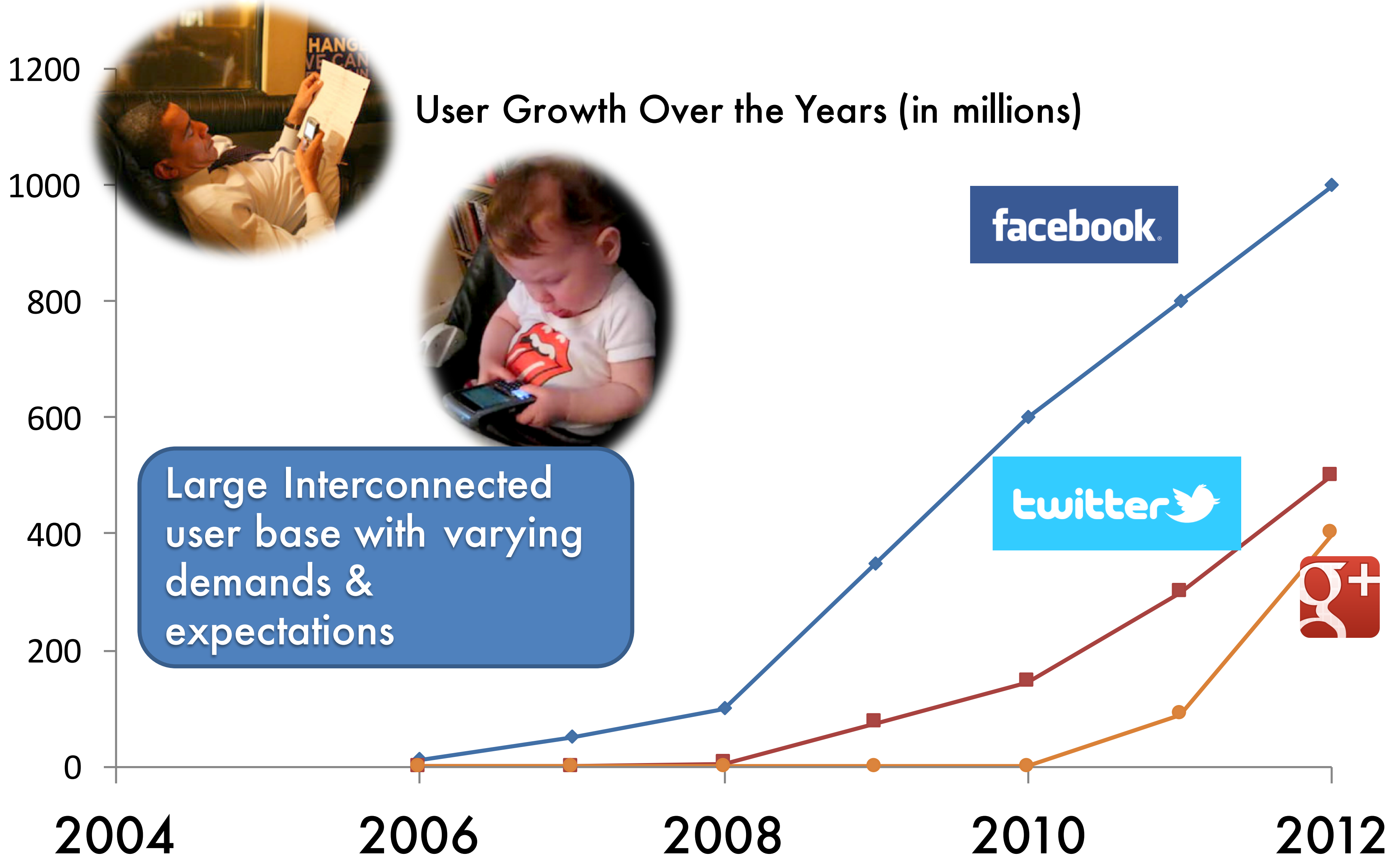
ebay

- – Over 1 billion page views per day
- – 44 billion SQL executions per day

facebook®

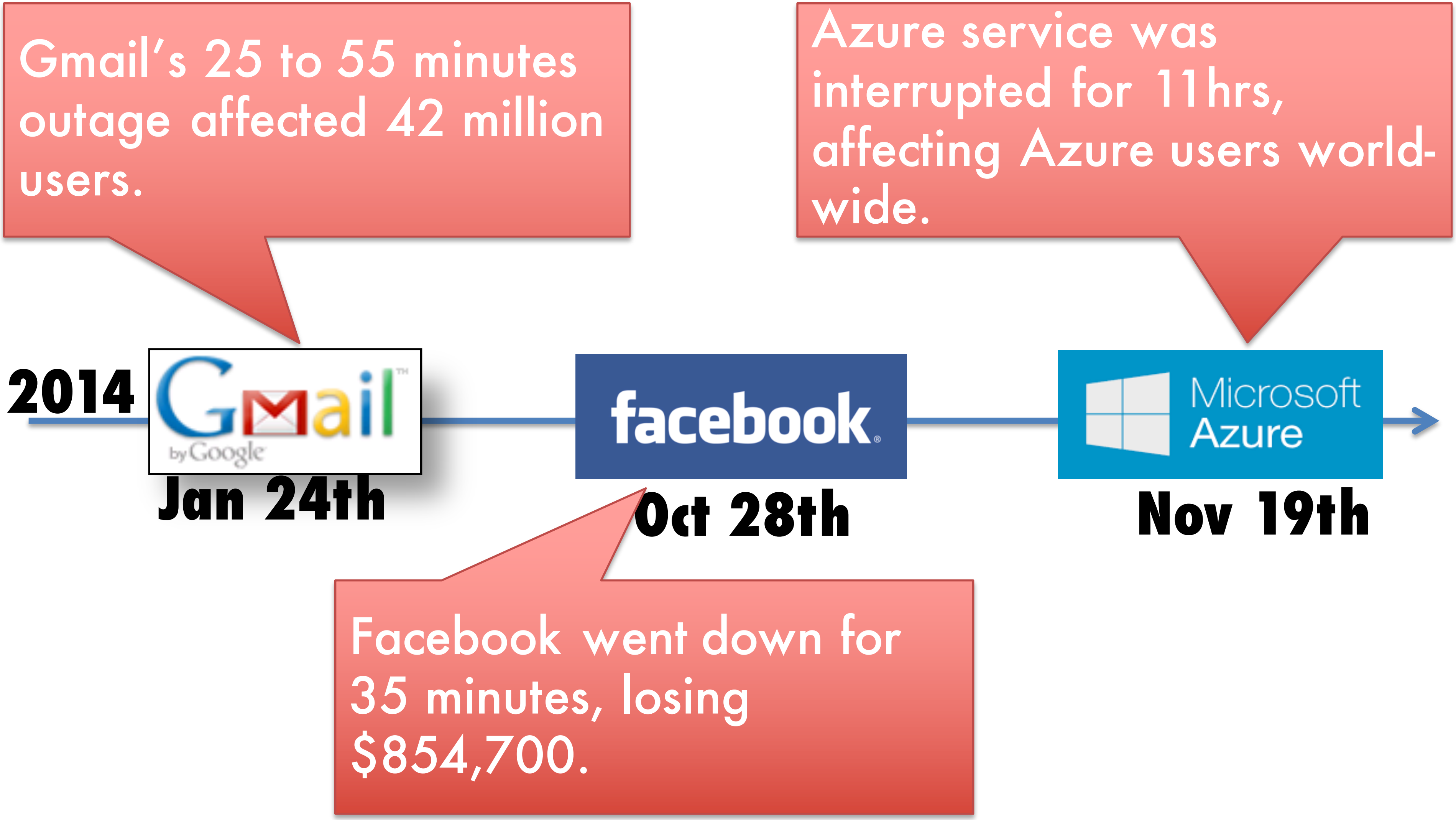
- – 8 billion minutes online everyday
- – Over 1.2 million photos a sec at peak

Rapid Growth and Evolution



Large Interconnected user base with varying demands & expectations

Quality of such systems is important



Gmail's 25 to 55 minutes outage affected 42 million users.

Azure service was interrupted for 11hrs, affecting Azure users world-wide.

2014

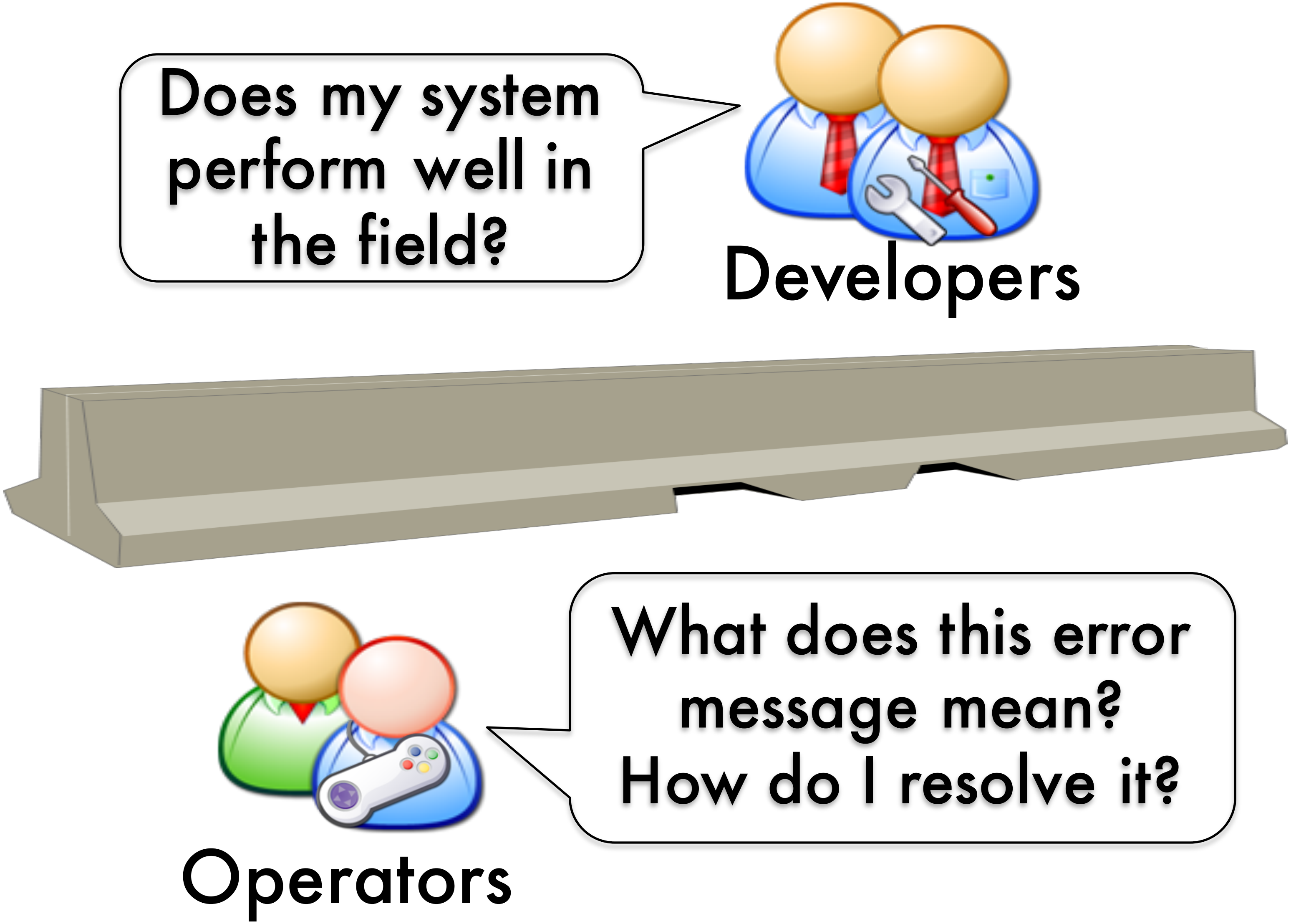
Jan 24th

Oct 28th

Nov 19th

Facebook went down for 35 minutes, losing $854,700.

There is a gap between software developers and operators



Does my system perform well in the field?

Developers

What does this error message mean? How do I resolve it?

Operators

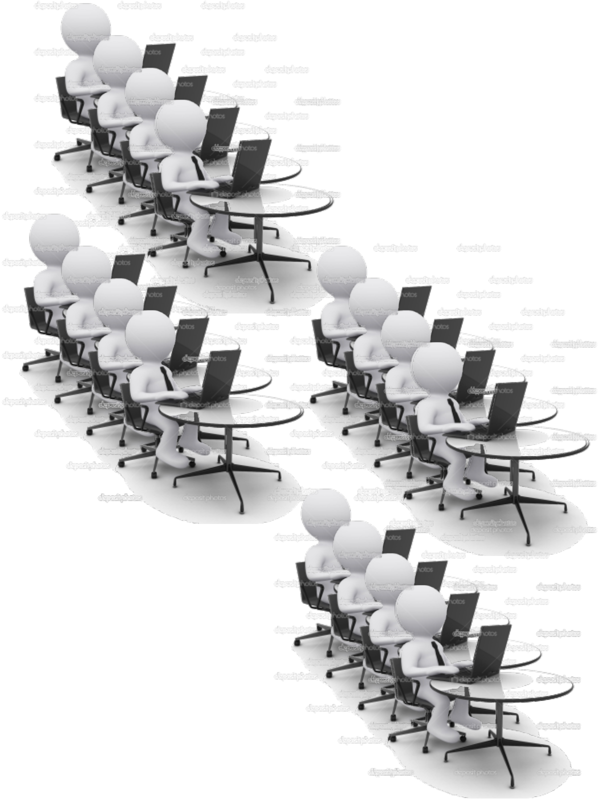
Discrepancy between development and deployment

Big data and
real-life environemnt

Small sample data AND
pseudo environment

"… move back and forth from local machines to cloud-based systems"

Microsoft® Research



How to ensure systems run correctly in the field?

SMALL SAMPLE DATA AND
PSEUDO ENVIRONMENT

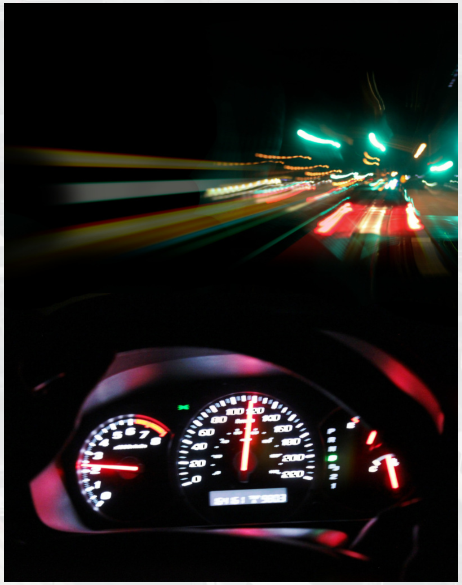
Running correctly

BIG DATA AND
REAL-LIFE ENVIRONEMNT

What exactly does this message mean?

Testing

What happens in the field



Filed issues

Higher intensity

Different feature usage

Very different workloads

As a result...

Risky deployments

It works on my machine!

Fear of change

How to release more reliable applications **faster** and more **frequently**?

The rapid release cycle of modern software systems

Google
facebook
NETFLIX
salesforce
LinkedIn

Often release several times in one day!

Nightly builds

Builds are often on a schedule:

- Typically, developers work during a day, committing their changes that fix bugs and add new features
- At night time, while developers are sleeping, a build is executed to produce deliverables with the day's changes
- QA teams can pick up that build the next day to test the new features and validate the bug fixes

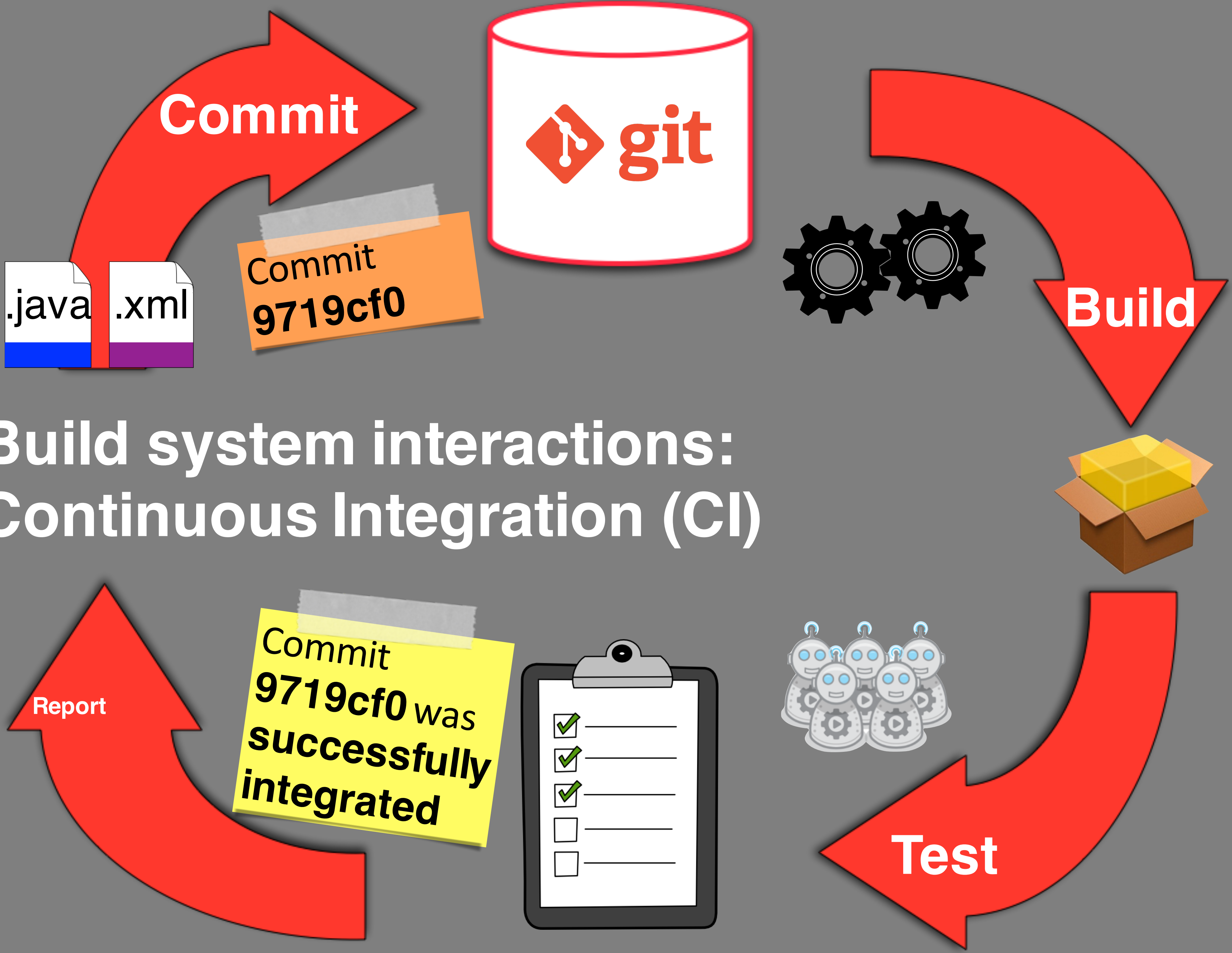
Build system interactions: The problem with nightly builds

Night builds are too infrequent:

- As the amount of change per day has grown, nightly builds have become difficult
- Consider the case when a nightly build does not complete cleanly
  - If hundreds of developers have committed changes, it’s hard to tell who caused the problem!
  - Imagine you broke the build, but you wrote the code yesterday! Hard to recall!

We need to run builds More frequently to keep up With fast-paced development!

Build system interactions: Continuous Integration (CI)



Commit

.java

.xml

Commit
9719cf0

git

Build

Test

Report

Commit
9719cf0 was
successfully
integrated

As a result…

Risky deployments

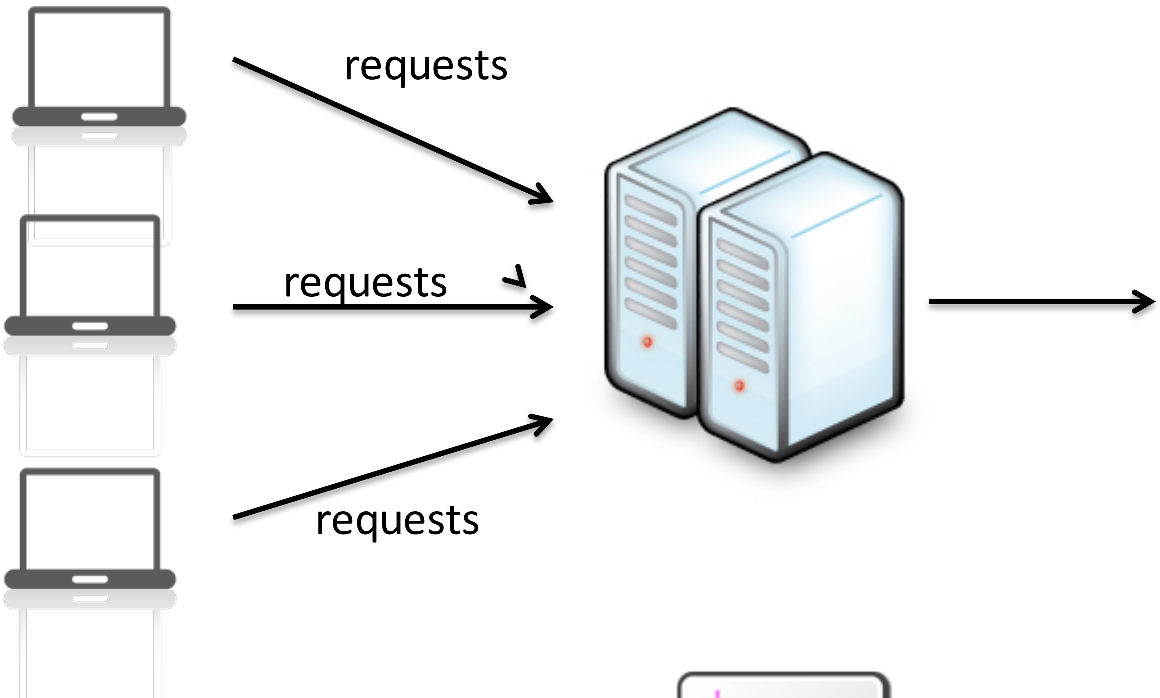
It works on my machine!

Fear of change

How to release more reliable applications **faster** and more **frequently**?

Leverage your data!

What data do we have?



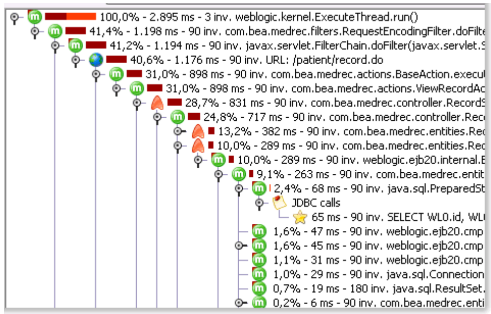
Crash report

Performance counters

Logs

Source Control

Issue tracking



```
100,0% - 2.895 ms - 3 inv. weblogic.kernel.ExecuteThread.run()
  41,4% - 1.198 ms - 90 inv. com.bea.medrec.filters.RequestEncodingFilter.doFilte
    41,2% - 1.194 ms - 90 inv. javax.servlet.FilterChain.doFilter(javax.servlet.S
      40,6% - 1.176 ms - 90 inv. URL: /patient/record.do
        31,0% - 898 ms - 90 inv. com.bea.medrec.actions.BaseAction.execu
          31,0% - 898 ms - 90 inv. com.bea.medrec.actions.ViewRecordAc
            28,7% - 831 ms - 90 inv. com.bea.medrec.controller.RecordS
              24,8% - 717 ms - 90 inv. com.bea.medrec.controller.Rec
                13,2% - 382 ms - 90 inv. com.bea.medrec.entities.Rec
                10,0% - 289 ms - 90 inv. com.bea.medrec.entities.Rec
                  10,0% - 289 ms - 90 inv. weblogic.ejb20.internal.E
                    9,1% - 263 ms - 90 inv. com.bea.medrec.entit
                      2,4% - 68 ms - 90 inv. java.sql.PreparedSt
                        JDBC calls
                          65 ms - 90 inv. SELECT WL0.id, WL
                      1,6% - 47 ms - 90 inv. weblogic.ejb20.cmp
                      1,6% - 45 ms - 90 inv. weblogic.ejb20.cmp
                      1,1% - 31 ms - 90 inv. weblogic.ejb20.cmp
                      1,0% - 29 ms - 90 inv. java.sql.Connection
                      0,7% - 19 ms - 180 inv. java.sql.ResultSet.
                      0,2% - 6 ms - 90 inv. com.bea.medrec.enti
```

Trace

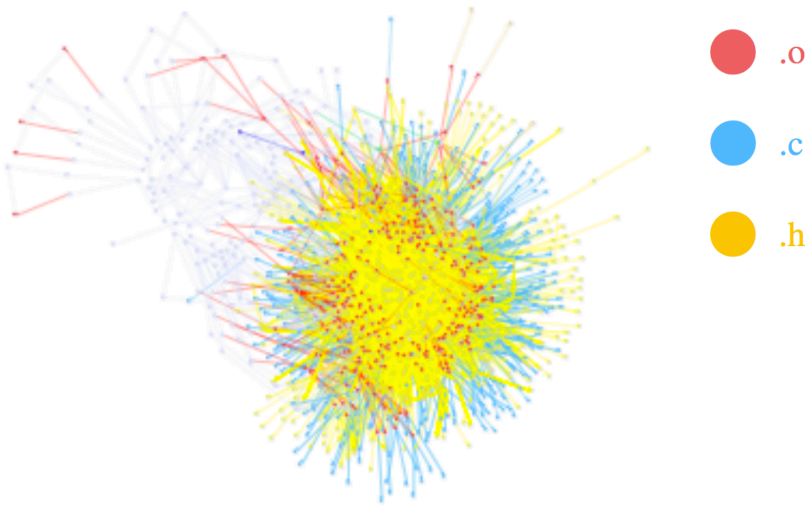
What kind of techniques can we learn from the class?

Statistical analysis
Data mining
Machine learning
Code analysis
...
More importantly:
How to conduct proper SE and System studies

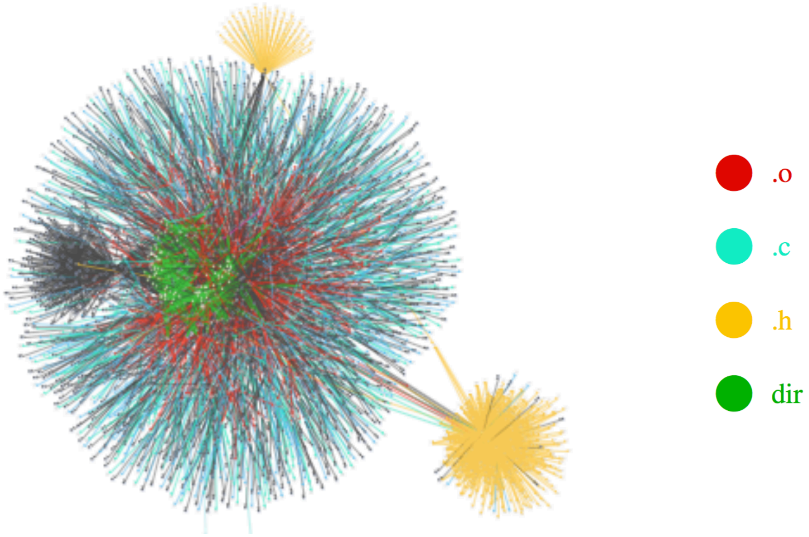
Help can these data help?

Can you give me several examples?

Build dependency graph



Linux 2.4



Linux 2.6

Bugs often repeat

Too Many
Connections!

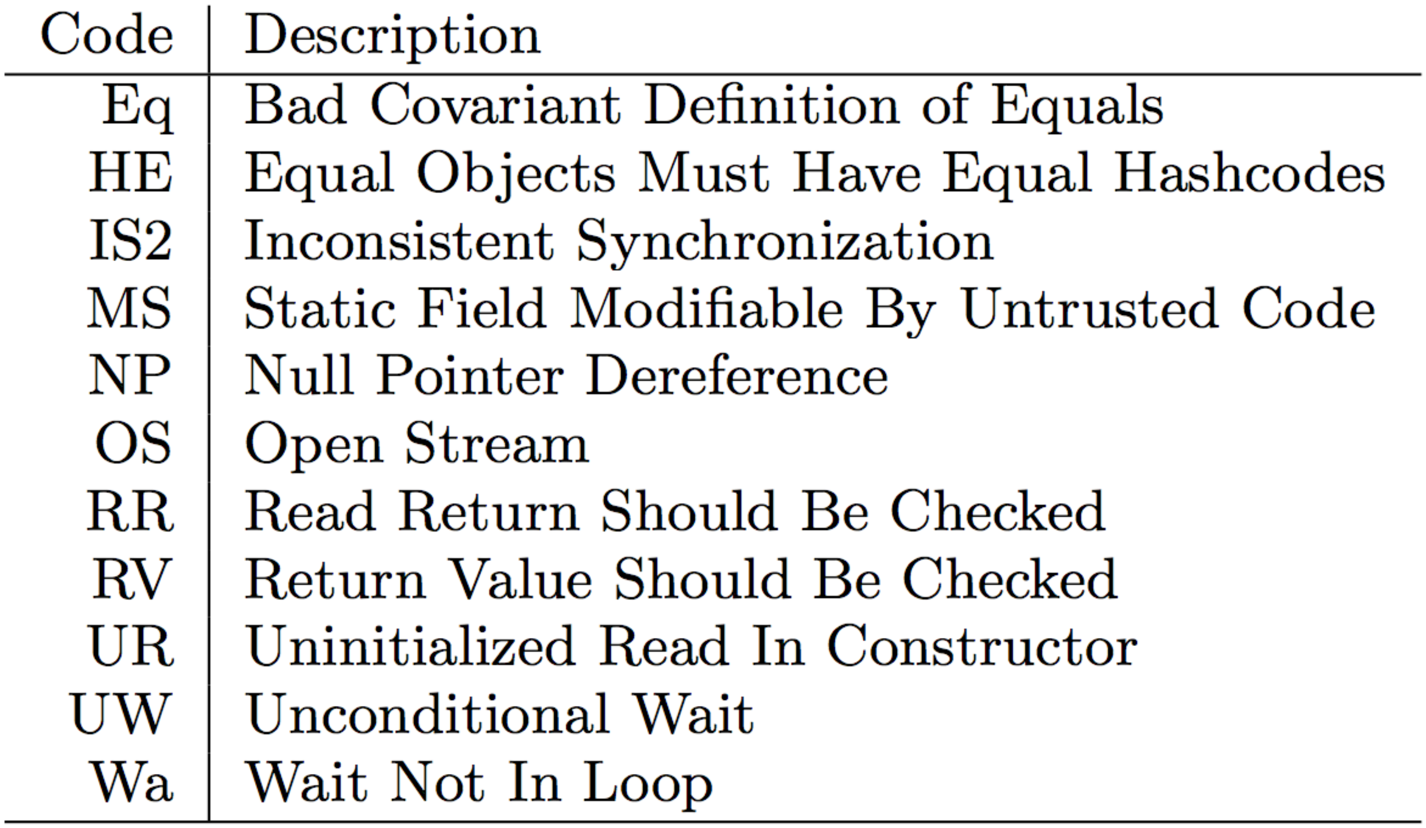
== ≠ =

What are the bugs in real world?

- Obvious/dumb bugs exist in real code.
  - while subtle and unique bugs exist, there are also many errors, even in production code, that are blatant, well-understood, and easy to find if you know what to look for.
- Because of the sheer complexity of modern object oriented languages like Java, the potential for misuse of language features and APIs is enormous

Simple pattern matching can find many bugs.

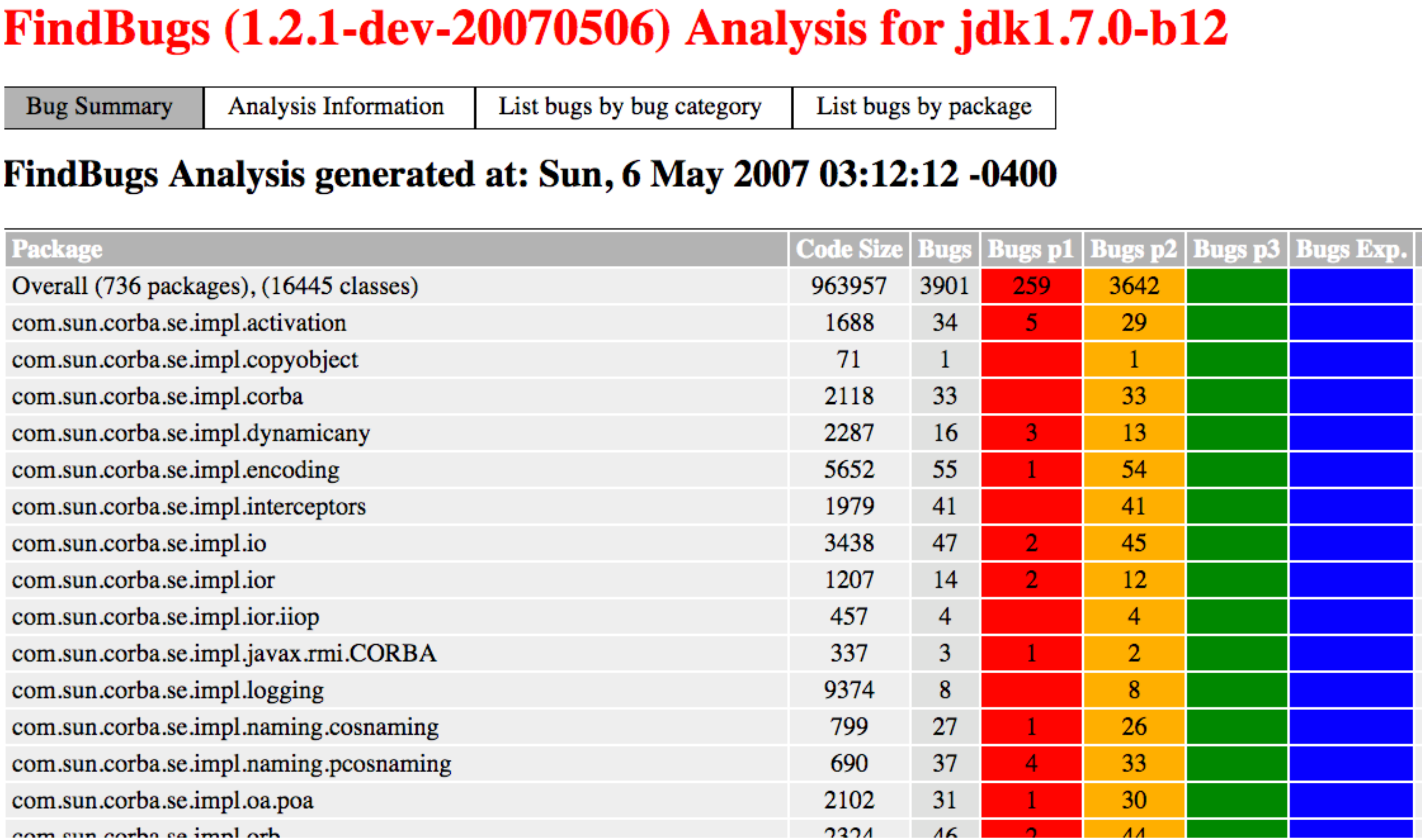
Generating bug patterns (examples)



Code	Description
Eq	Bad Covariant Definition of Equals
HE	Equal Objects Must Have Equal Hashcodes
IS2	Inconsistent Synchronization
MS	Static Field Modifiable By Untrusted Code
NP	Null Pointer Dereference
OS	Open Stream
RR	Read Return Should Be Checked
RV	Return Value Should Be Checked
UR	Uninitialized Read In Constructor
UW	Unconditional Wait
Wa	Wait Not In Loop

A longer list from FindBugs:
http://findbugs.sourceforge.net/bugDescriptions.html

FindBugs results on JDK1.7

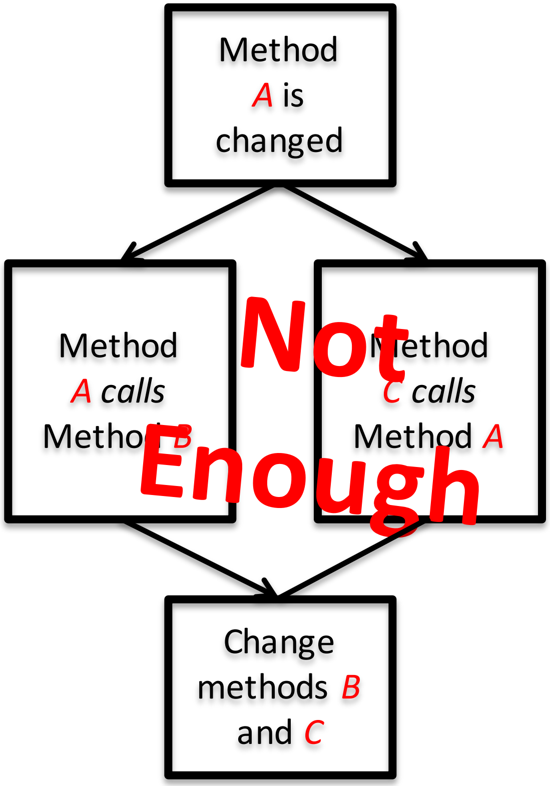


FindBugs (1.2.1-dev-20070506) Analysis for jdk1.7.0-b12

FindBugs Analysis generated at: Sun, 6 May 2007 03:12:12 -0400

Package	Code Size	Bugs	Bugs p1	Bugs p2	Bugs p3	Bugs Exp.
Overall (736 packages), (16445 classes)	963957	3901	259	3642		
com.sun.corba.se.impl.activation	1688	34	5	29		
com.sun.corba.se.impl.copyobject	71	1		1		
com.sun.corba.se.impl.corba	2118	33		33		
com.sun.corba.se.impl.dynamicany	2287	16	3	13		
com.sun.corba.se.impl.encoding	5652	55	1	54		
com.sun.corba.se.impl.interceptors	1979	41		41		
com.sun.corba.se.impl.io	3438	47	2	45		
com.sun.corba.se.impl.ior	1207	14	2	12		
com.sun.corba.se.impl.ior.iiop	457	4		4		
com.sun.corba.se.impl.javax.rmi.CORBA	337	3	1	2		
com.sun.corba.se.impl.logging	9374	8		8		
com.sun.corba.se.impl.naming.cosnaming	799	27	1	26		
com.sun.corba.se.impl.naming.pcosnaming	690	37	4	33		
com.sun.corba.se.impl.oa.poa	2102	31	1	30		
com.sun.corba.se.impl.orb	2324	46	2	44		

Propagating code changes



Method *A* is changed

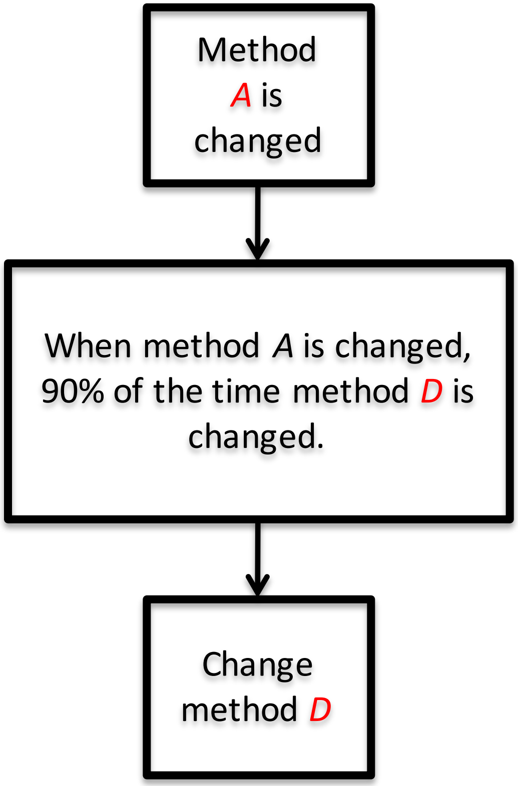
Method *A calls* Method *B*

Method *C calls* Method *A*

Change methods *B* and *C*

Not Enough

History helps!



Method *A* is changed

When method *A* is changed, 90% of the time method *D* is changed.

Change method *D*

Should I test\review my?

A. Ten *most-complex* functions

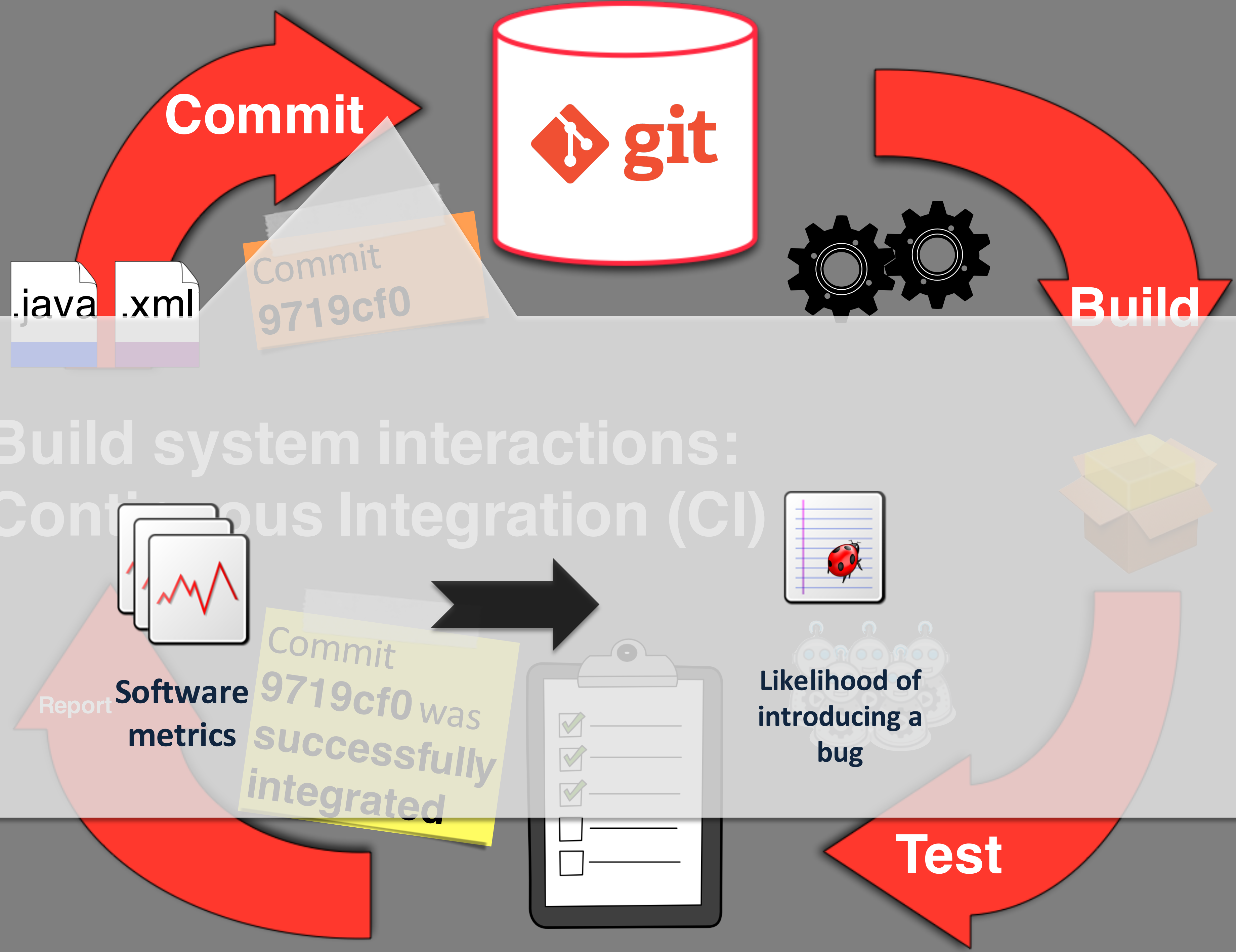
B. Ten *largest* functions

[✓ C. Ten *most-fixed* functions]

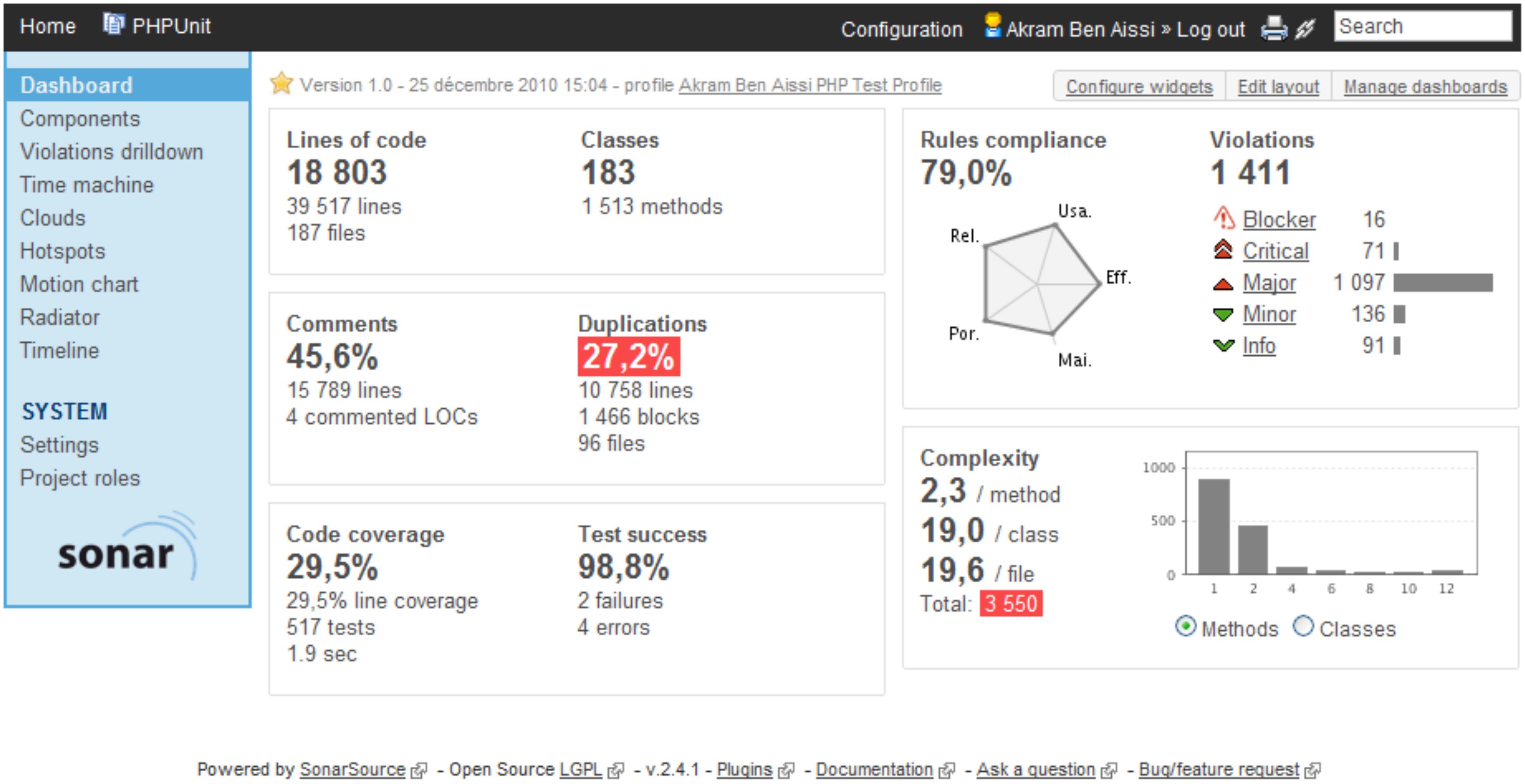
Who produces more buggy code?

A. Junior Developer [✓ B. Senior Developer]

Build system interactions: Continuous Integration (CI)



sonarsource™



Home | PHPUnit | Configuration | Akram Ben Aissi » Log out | Search

Dashboard
- Components
- Violations drilldown
- Time machine
- Clouds
- Hotspots
- Motion chart
- Radiator
- Timeline

SYSTEM
- Settings
- Project roles

sonar

Version 1.0 - 25 décembre 2010 15:04 - profile Akram Ben Aissi PHP Test Profile

Configure widgets | Edit layout | Manage dashboards

Lines of code
18 803
39 517 lines
187 files

Classes
183
1 513 methods

Comments
45,6%
15 789 lines
4 commented LOCs

Duplications
27,2%
10 758 lines
1 466 blocks
96 files

Code coverage
29,5%
29,5% line coverage
517 tests
1.9 sec

Test success
98,8%
2 failures
4 errors

Rules compliance
79,0%

Violations
1 411

Blocker	16
Critical	71
Major	1 097
Minor	136
Info	91

Complexity
2,3 / method
19,0 / class
19,6 / file
Total: 3 550

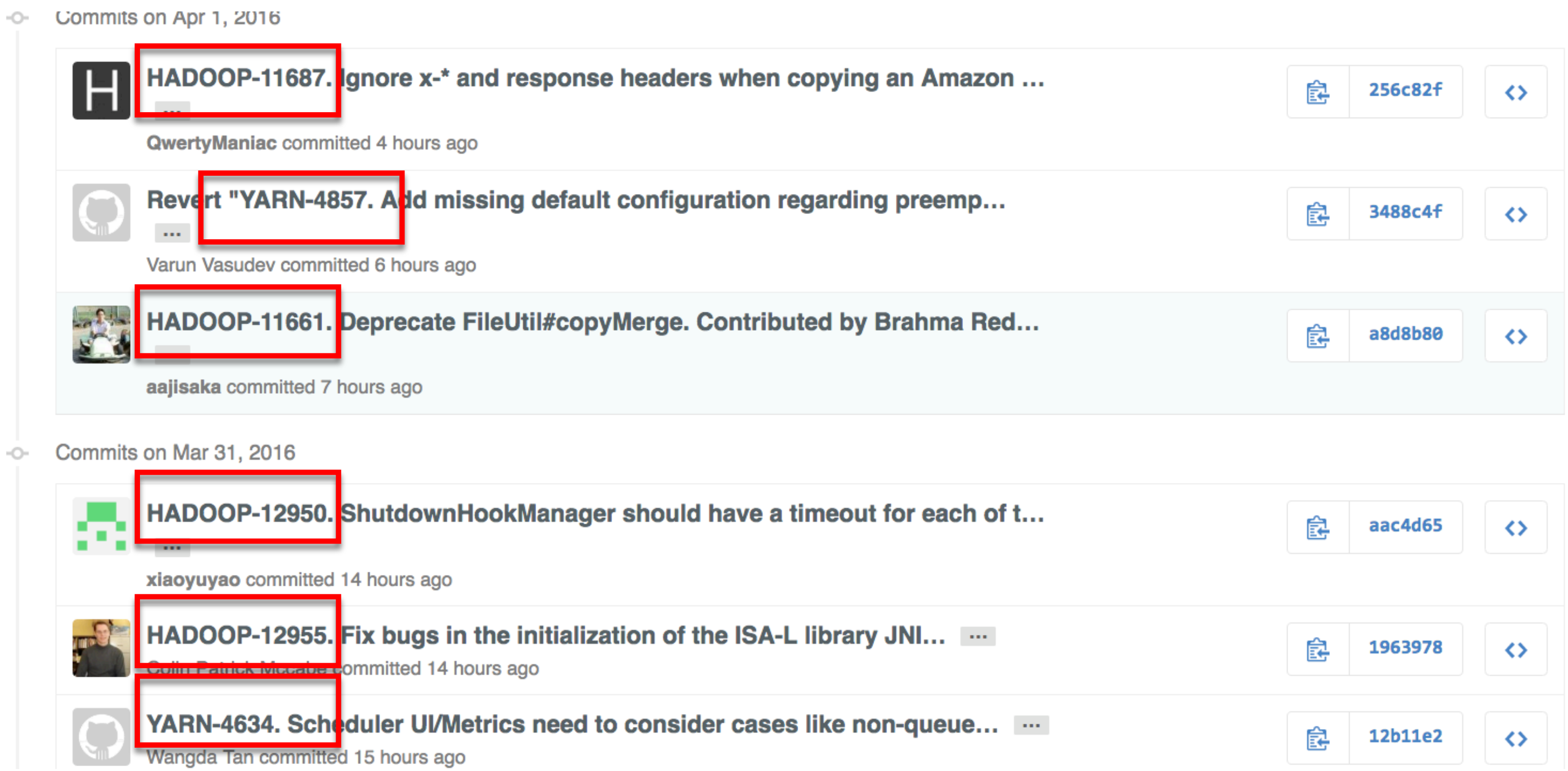
● Methods ○ Classes

Powered by SonarSource - Open Source LGPL - v.2.4.1 - Plugins - Documentation - Ask a question - Bug/feature request

Chicken Versus Egg Problem

Practitioners are not willing to improve repository data till they see value

Some practices have become convention



Commits on Apr 1, 2016

HADOOP-11687. Ignore x-* and response headers when copying an Amazon … …
QwertyManiac committed 4 hours ago
256c82f

Revert "YARN-4857. Add missing default configuration regarding preemp… …
Varun Vasudev committed 6 hours ago
3488c4f

HADOOP-11661. Deprecate FileUtil#copyMerge. Contributed by Brahma Red…
aajisaka committed 7 hours ago
a8d8b80

Commits on Mar 31, 2016

HADOOP-12950. ShutdownHookManager should have a timeout for each of t… …
xiaoyuyao committed 14 hours ago
aac4d65

HADOOP-12955. Fix bugs in the initialization of the ISA-L library JNI… …
Colin Patrick McCabe committed 14 hours ago
1963978

YARN-4634. Scheduler UI/Metrics need to consider cases like non-queue… …
Wangda Tan committed 15 hours ago
12b11e2

Including Issue ID in commit comments

Detecting performance regression

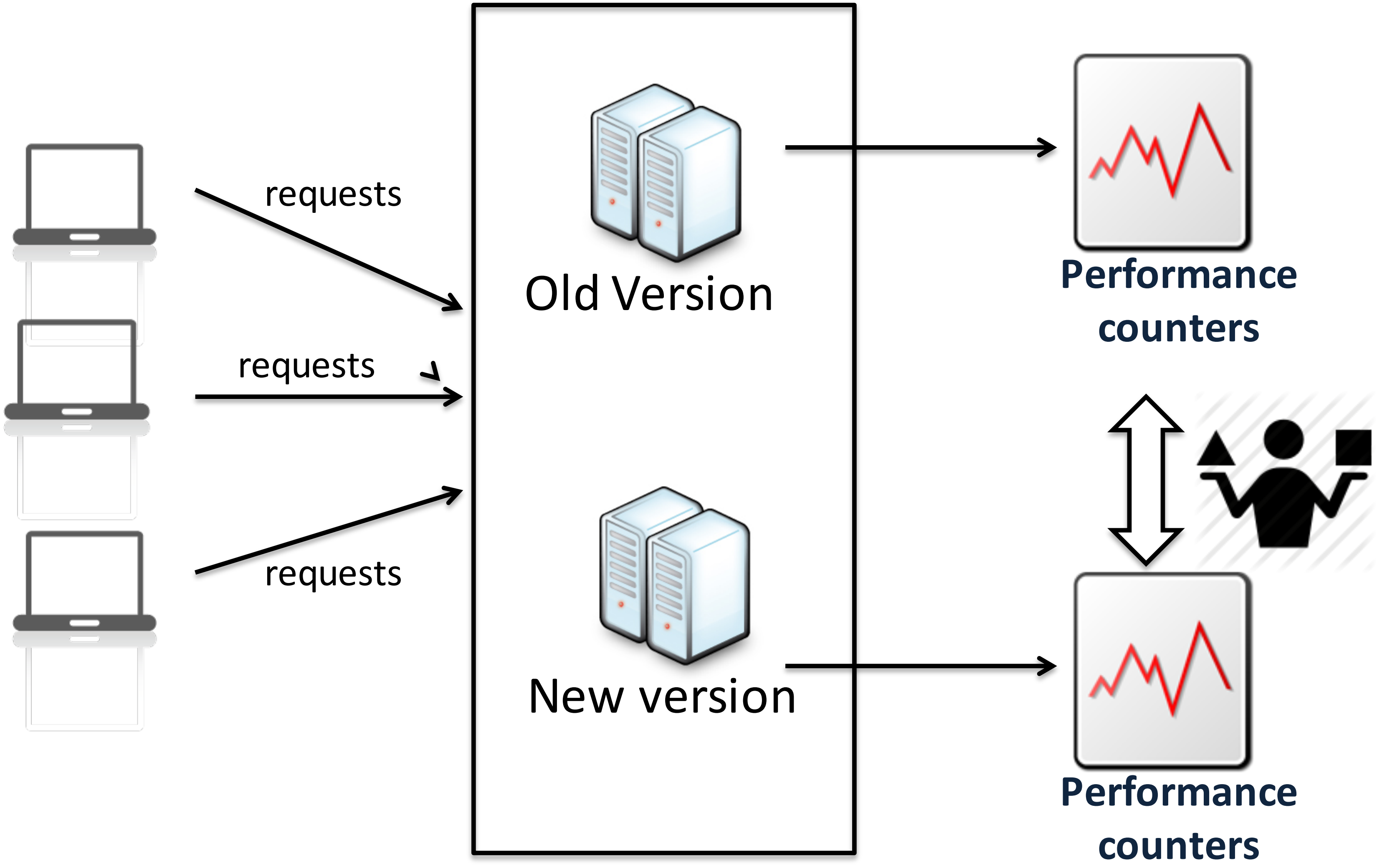
What is a performance regression?

Old

rsion

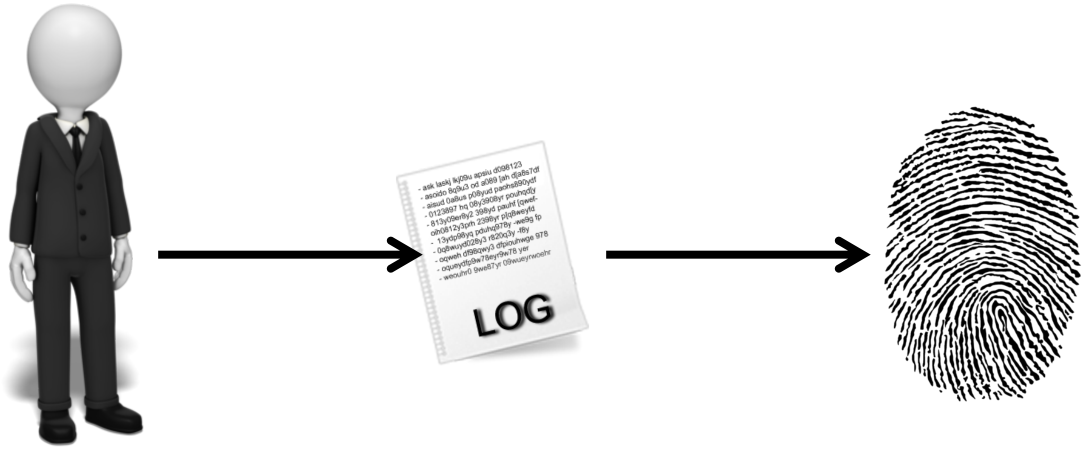
Does the new version have worse performance than the old version?

How to detect performance regression?



Are you testing realisticaly?

We can compare field and test workloads using logs



Is the behavior of this person covered in testing?

Understanding error messages

Practitioners have challenges in understanding log lines

Fetch failure

What exactly does this message mean?

What could be the cause?

Is it affecting my data?

Looking for an expert is not the optimal approach to resolve log inquiries

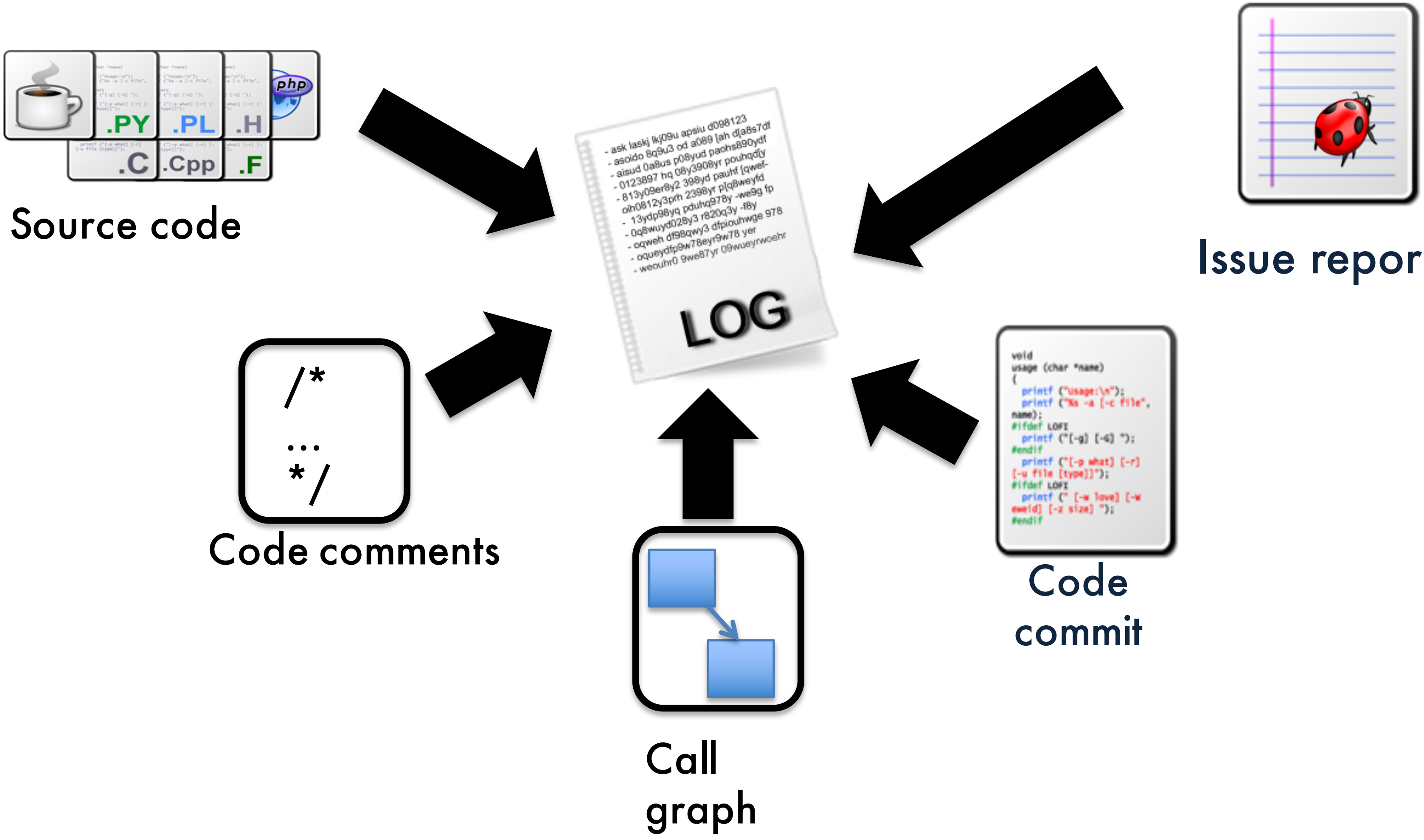
Identifying the expert of a log line is challenging.

Wrong answers may be posted in reply to inquiries.

Over 20% of the inquires have no reply.

First reply can take up to 210 hours.

Attach development knowledge to logs



Source code

```
- ask laskj lkj09u apsiu d098123
- asoido 8q9u3 od a089 [ah d[a8s7df
- aisud 0a8us p08yud paohs890ydf
- 0123897 hq 08y3908yr pouhqdfy
- 813y09er8y2 398yd pauhf [qwef-
oih0812y3prh 2398yr p[q8weyfd
-  13ydp98yq pduhq978y -we9g fp
- 0q8wuyd028y3 r820q3y -f8y
- oqweh df98qwy3 dfpiouhwge 978
- oqueydfp9w78eyr9w78 yer
- weouhr0 9we87yr 09wueyrwoehr
```

Issue reports

Code comments

```
/*
...
*/
```

Call graph

```
void
usage (char *name)
(
  printf ("Usage:\n");
  printf ("%s -a [-c file",
name);
#ifdef LOFI
  printf ("[-g] [-G] ");
#endif
  printf ("[-p what] [-r]
[-u file [type]]");
#ifdef LOFI
  printf (" [-w love] [-w
eweld] [-z size] ");
#endif
```

Code commit

How can these data help?

More will be covered in the class later.

How will I be **evaluated**?

How will I be evaluated?

Paper presentation and discussion (20%): 10% as presenter+5% as discussant+5% activity in class

Each group (2 people) acts as presenter once and discussant once in a term. Audience randomly picked for summary. You need to read ALL papers.

How will I be evaluated?

Weekly paper critique (10%)

5 weeks in total (since there is one week for presentation).
Done individually.
Done over Easychair.
Submitted before Tuesday.

How will I be evaluated?

Assignment (20%):
Including developing a code analysis and metrics extraction tool.
3 page report in IEEE format+submitting the source code+executable.

Details covered in week2.

How will I be evaluated?

Project (50%): 10% project update+20% final report+20%

Topics: paper replication, or any other topics lated to the class

Project proposal: no grade, just for help

Project update: 10 minutes presentation

Project presentation: 15 minutes 20%

Project report: 20% 10 pages IEEE format

Where are the course mateirals?

Course website: http://users.encs.concordia.ca/~shang/soen691/current

More importantly EasyChair

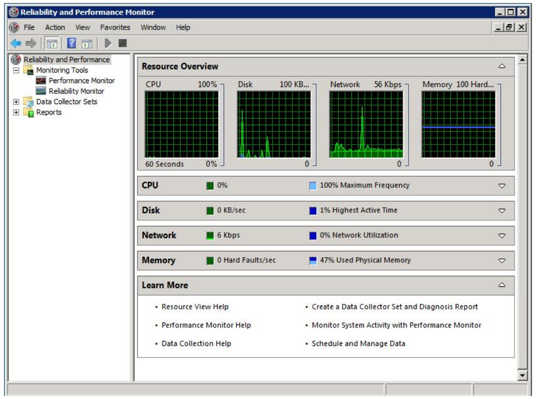
The conference system

Challenges of mining large software data for DevOps

Weiyi Shang

UNIVERSITÉ Concordia UNIVERSITY

How to monitor ULSS with minimal overhead?



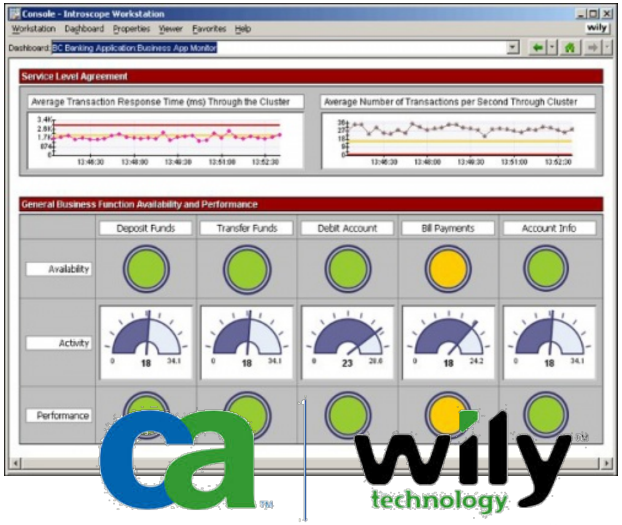
CPU 0% 100% Maximum Frequency

Disk 0 KB/sec 1% Highest Active Time

Network 6 Kbps 0% Network Utilization

Memory 0 Hard Faults/sec 47% Used Physical Memory

- Resource View Help
- Performance Monitor Help
- Data Collection Help
- Create a Data Collector Set and Diagnosis Report
- Monitor System Activity with Performance Monitor
- Schedule and Manage Data

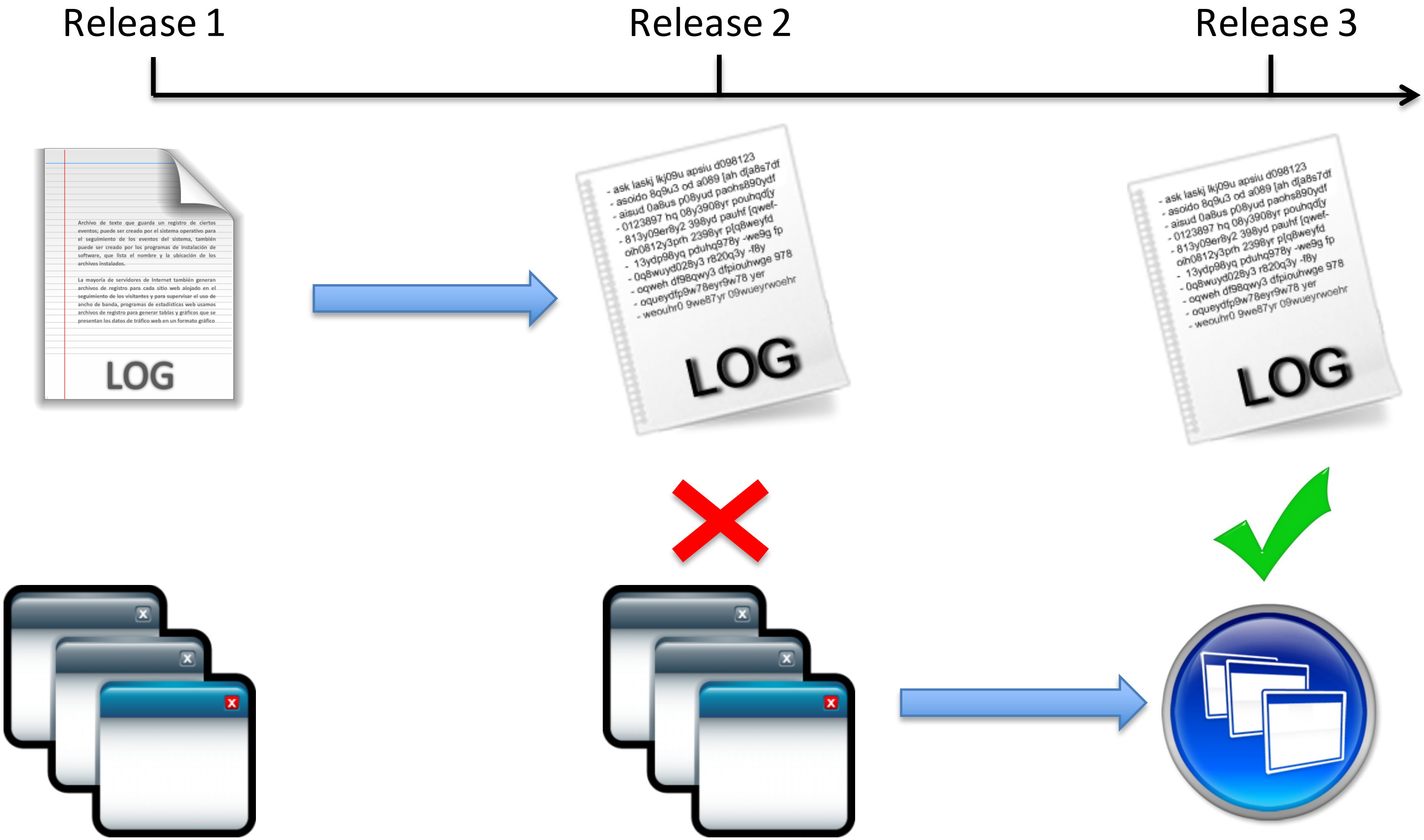


Dashboard: BC Banking Application Business App Monitor

More frequent monitor

Overhead

How to better leverage logs in practice?

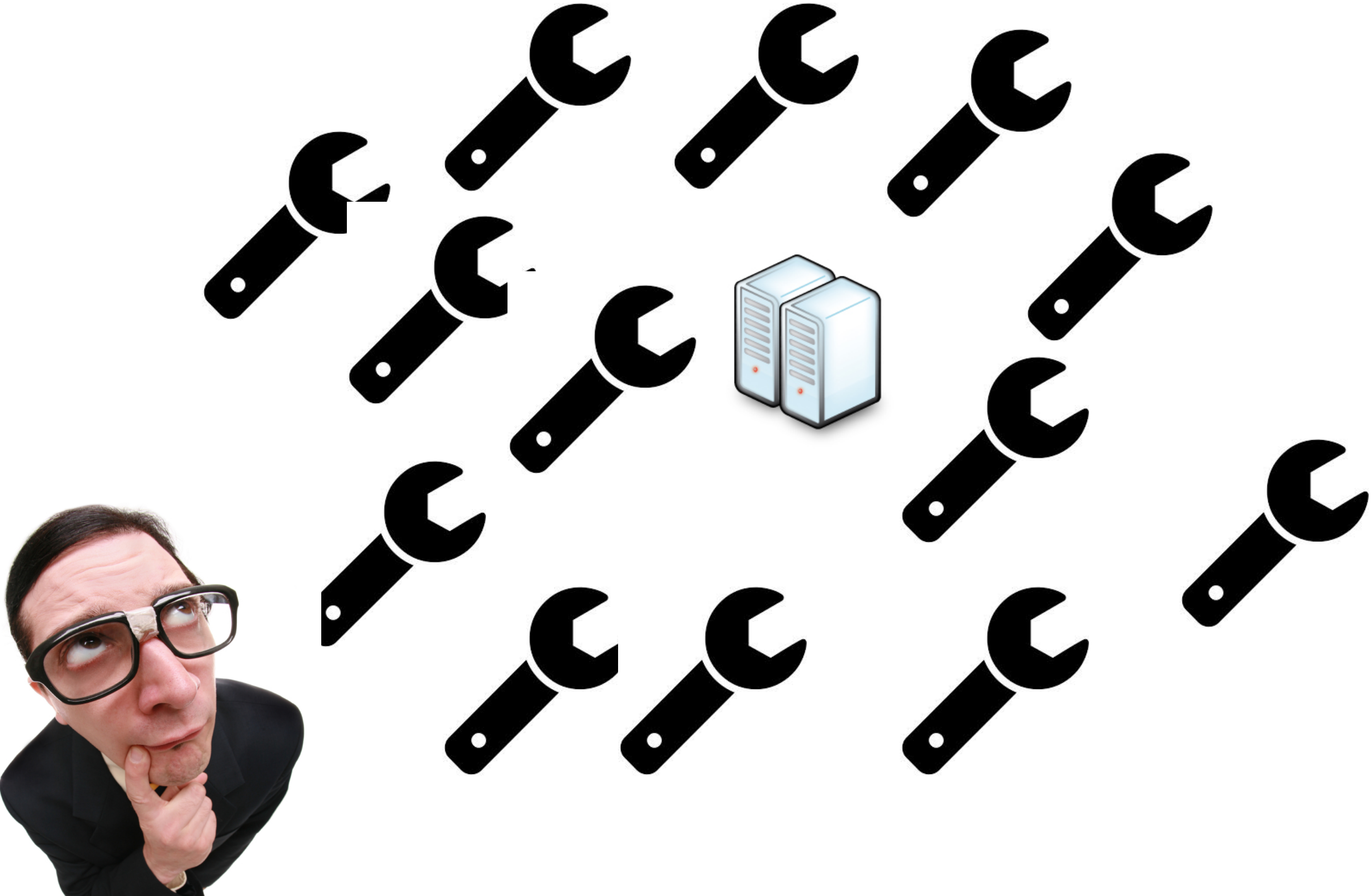


Release 1

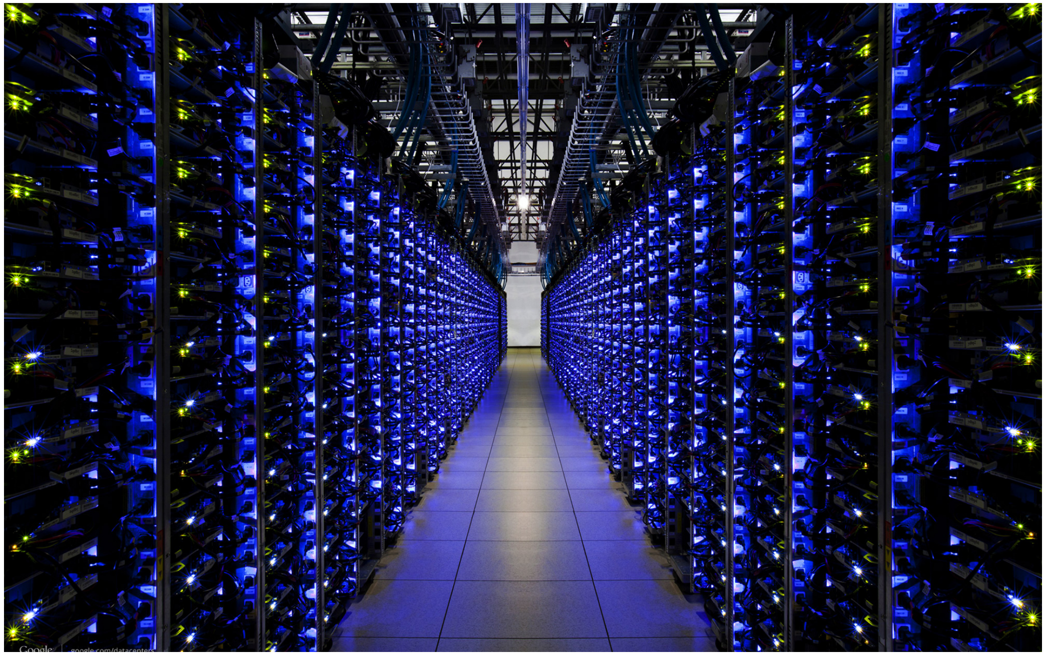
Release 2

Release 3

How to ensure optimal configuration?



How to model and save power consumption?



Large software systems generate large amounts of performance counters