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

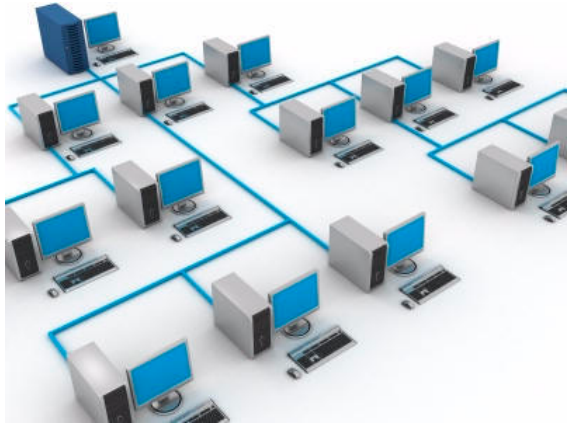
Tse-Hsun (Peter) Chen

UNIVERSITÉ Concordia UNIVERSITY

Who is this guy?

Prof. Chen's research

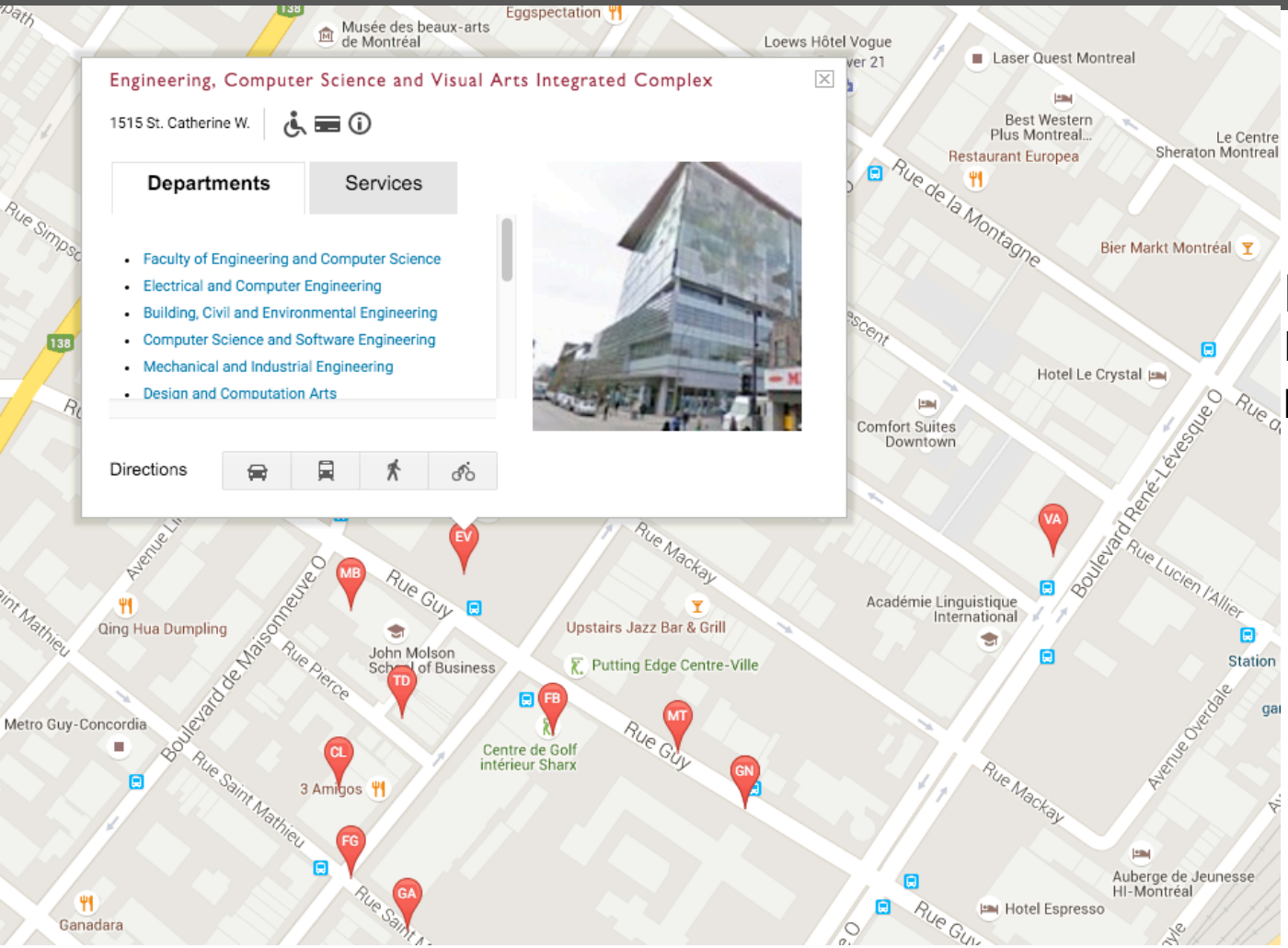
Performance of database-centric systems



Software engineering for large systems

Mining software data

Office



Room # EV3.249

Email:
peterc@encs.concordia.ca

Contact

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

TA: Maxime Lamothe
email: lamothe.max@gmail.com

What do I need to do to survive?

What do I need to survive?

This is NOT a lecture course!

- Good discussion; express your opinions.
- Read papers.
- A good project.

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 paper summary and discussion.
- You need to read ALL papers.

How will I be evaluated?

- Paper reviews (10%)
- Submit ONE paper critique each week
  - 5 reviews in total (since there is one week for presentation).
  - Done individually.
  - Done over EasyChair.
- Submit summary for ALL papers
- To be submitted before Friday.

How will I be evaluated?

- Assignment (20%):
  - Including developing a code analysis and metrics extraction tool.
  - 5 page report in ACM format
    - submitting the source code AND executable.
  - Details covered in week 2.
  - Done in groups of 3~4 people.

How will I be evaluated?

Project (50%):

- 10% project update
- 20% project presentation
- 20% final report

Project proposal: no grade, just for help
Project update: 10 minutes presentation
Project presentation: 15 minutes
Project report: 10 pages ACM format

How will I be evaluated?

- Topics of the project:
  - Paper replication.
  - Any other topics related to the course.
- Done in groups of ~4
  - Can also be done individually

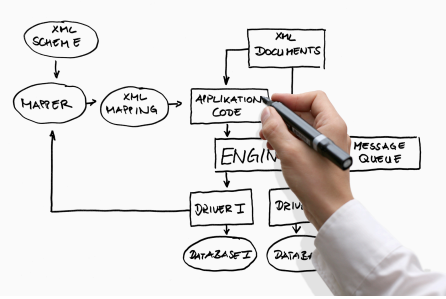
Where are the course materials?

Course website:
http://petertsehsun.github.io/soen691/current/

More importantly

What is DevOps?

Software Development

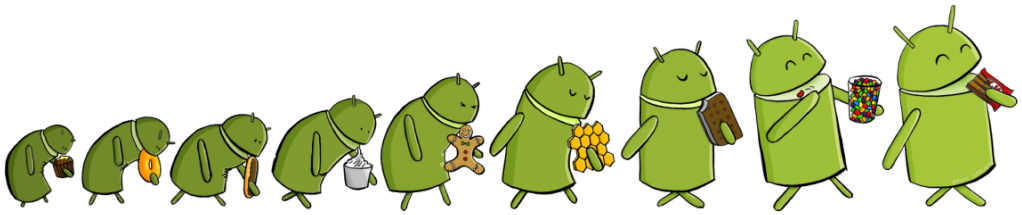


Design and specification

Coding

Testing

Release engineering

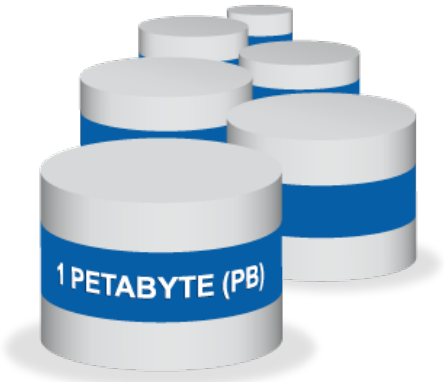


Evolution

Software Operation

Monitoring

Troubleshooting



Capacity planning

Anomaly detection

Q&A

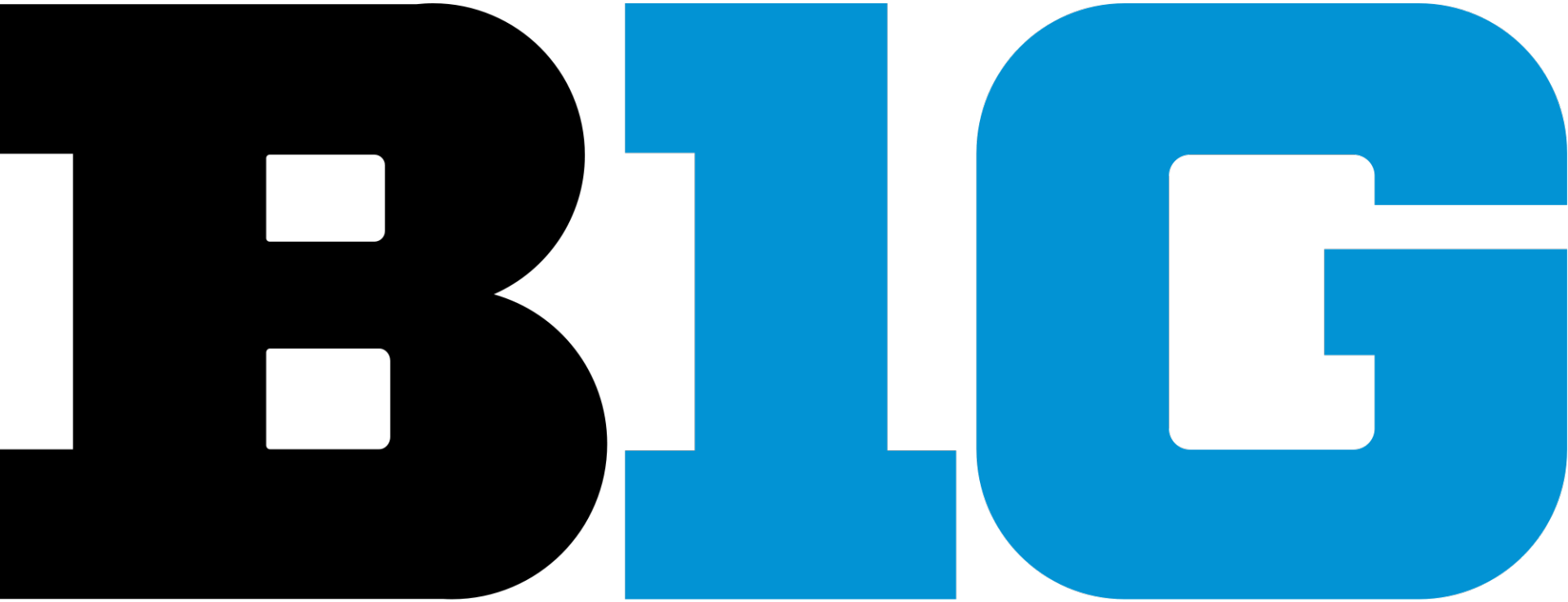
Configuration Tuning

What is DevOps?

Development (Dev) + Operations (Ops)

"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."

Context of DevOps



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

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

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

Rapid Growth and Evolution

User Growth Over the Years (in millions)

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.

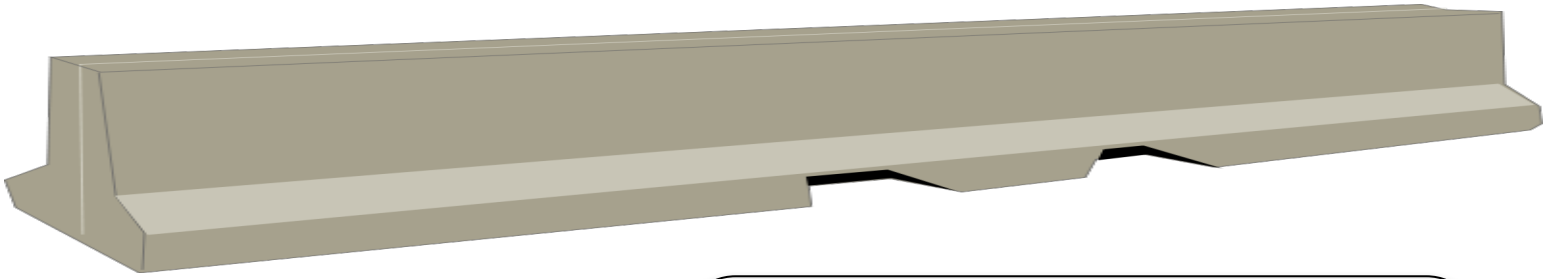
Azure service was interrupted for 11hrs, affecting Azure users world-wide.

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

How to ensure systems run correctly in the field?

Small sample data and pseudo environment

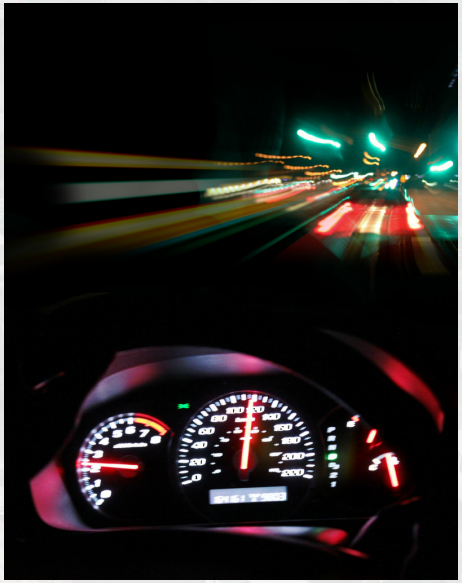
Running correctly

Big data and real-life environemnt

What exactly does this message mean?

Testing

What happens in the field



Field 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

Often release several times in one day!

Old approach: Nightly builds

Builds are often on a schedule:

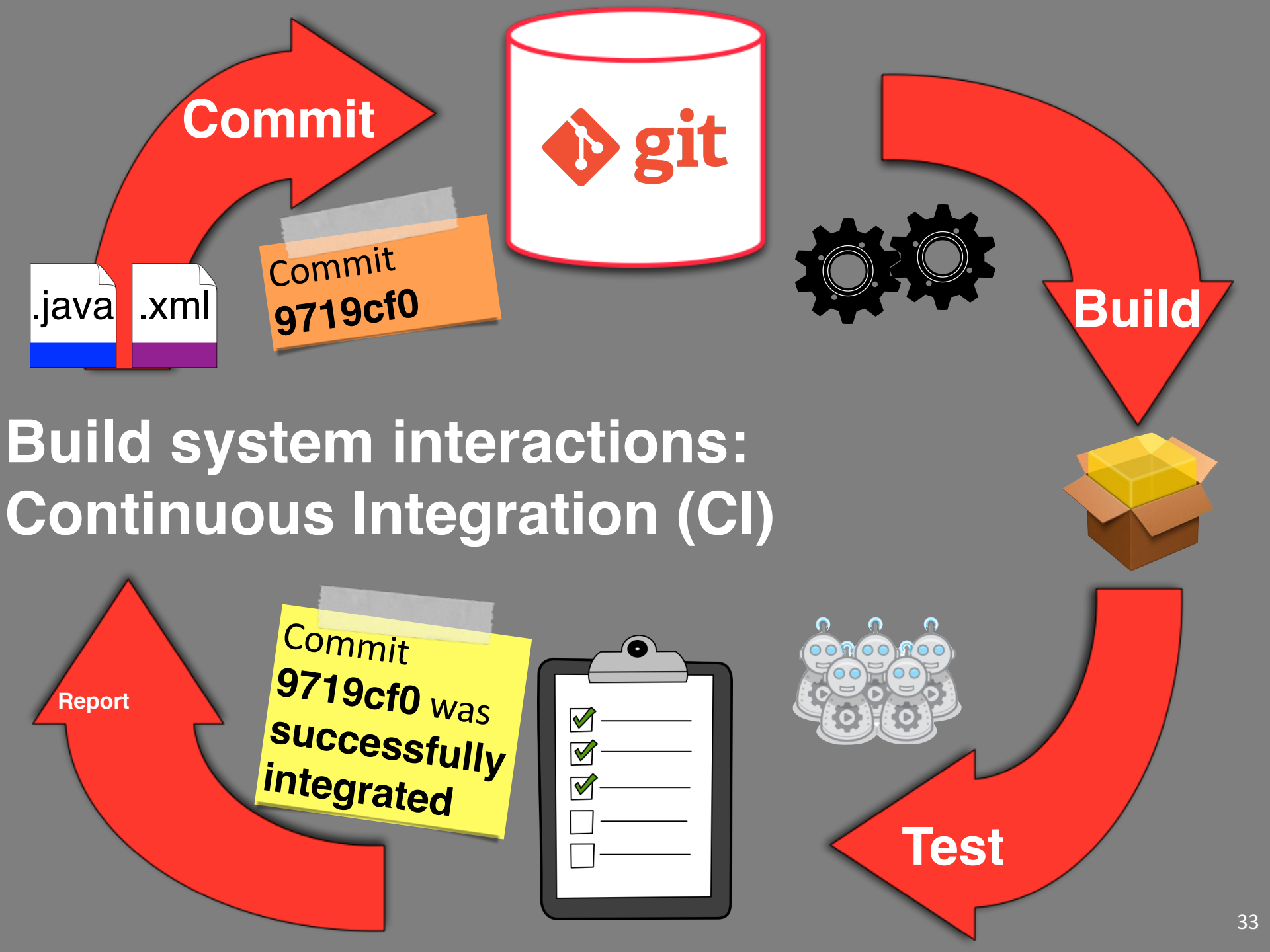
- Typically, developers work during a day, committing their changes that fix bugs and add new features
- At night time, while developers are sleeping, a build is executed to produce deliverables with the day's changes
- QA teams can pick up that build the next day to test the new features and validate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

Build system interactions: The problem with nightly builds

Night builds are too infrequent:

- As the amount of changes per day has grown, nightly builds have become difficult
- Consider the case when a nightly build does not complete correctly
- 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)

As a result…

Risky deployments

It works on my machine!

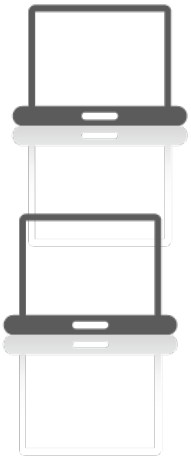
Fear of change

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

Leverage your data!

What data do we have?

Crash report



requests

requests

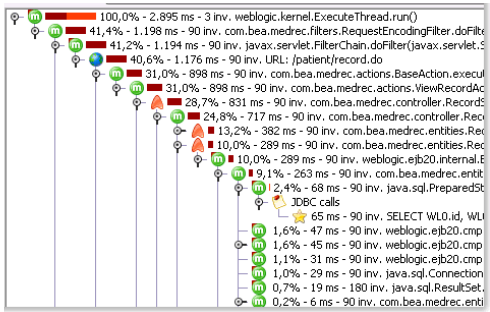
requests

Performance counters

Logs

Source Control

Issue tracking



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

What are we doing here?

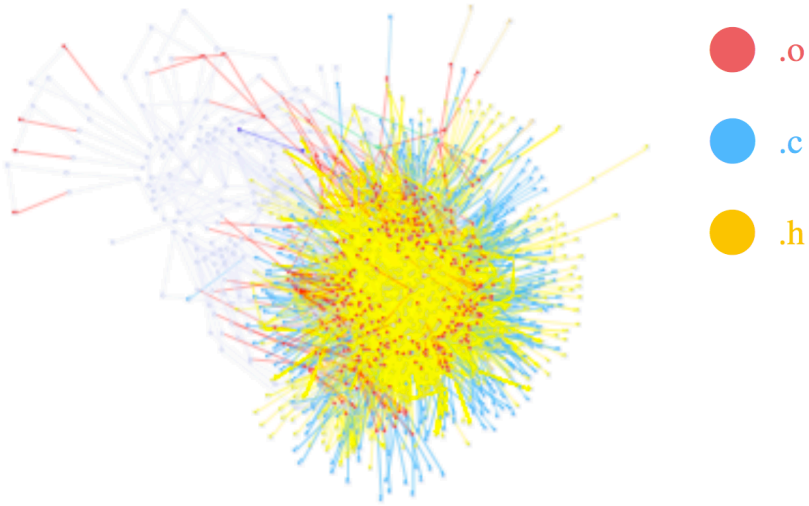
Learning about the how to leverage the data in software systems in order to assist in DevOps.

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

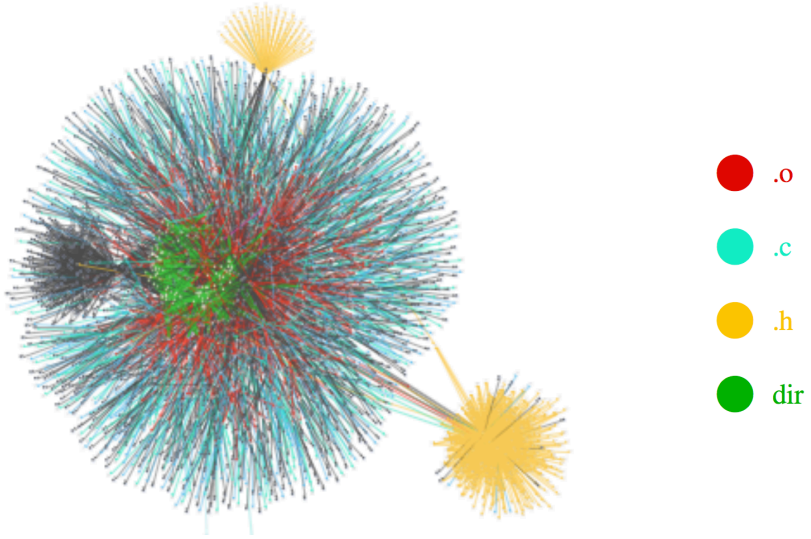
How can these data help?

Can you give me several examples?

Build dependency graph



Linux 2.4



Linux 2.6

Bugs often repeat

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

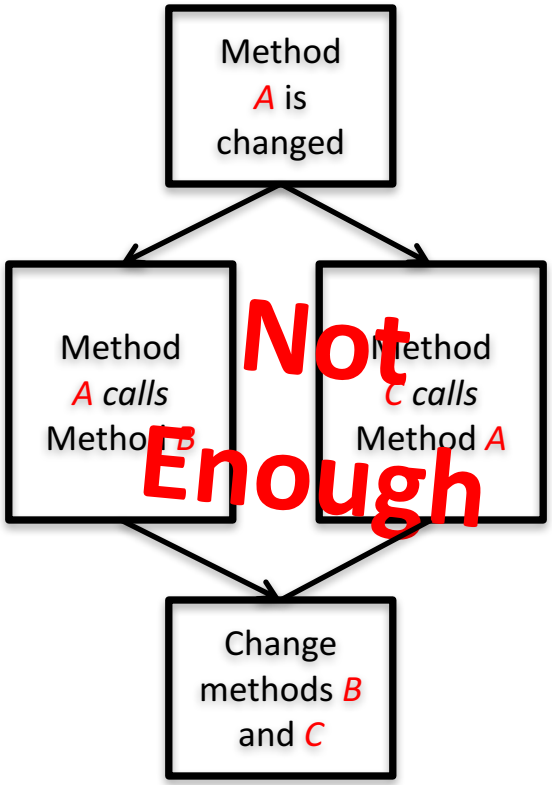
FindBugs (1.2.1-dev-20070506) Analysis for jdk1.7.0-b12

Bug Summary	Analysis Information	List bugs by bug category	List bugs by package

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

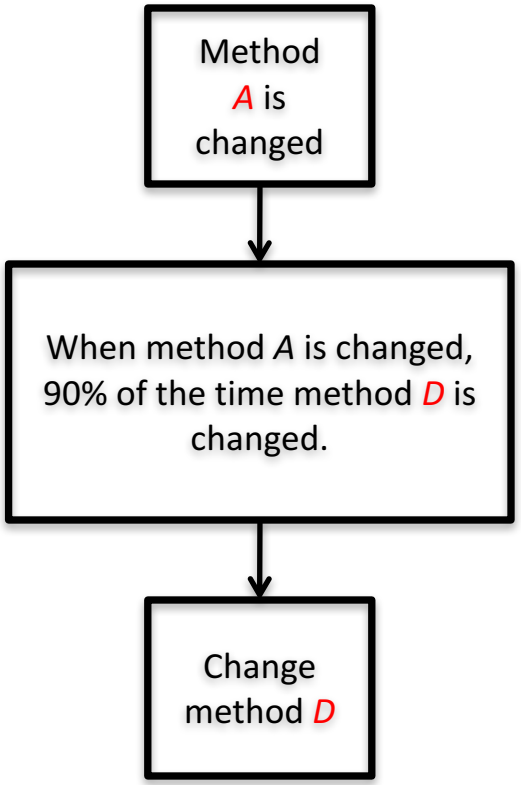
Package	Code Size	Bugs	Bugs p1	Bugs p2	Bugs p3	Bugs Exp.
Overall (736 packages), (16445 classes)	963957	3901	259	3642		
com.sun.corba.se.impl.activation	1688	34	5	29		
com.sun.corba.se.impl.copyobject	71	1		1		
com.sun.corba.se.impl.corba	2118	33		33		
com.sun.corba.se.impl.dynamicany	2287	16	3	13		
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



Not Enough

History helps!



Should I test\review my?

A. Ten *most-complex* functions

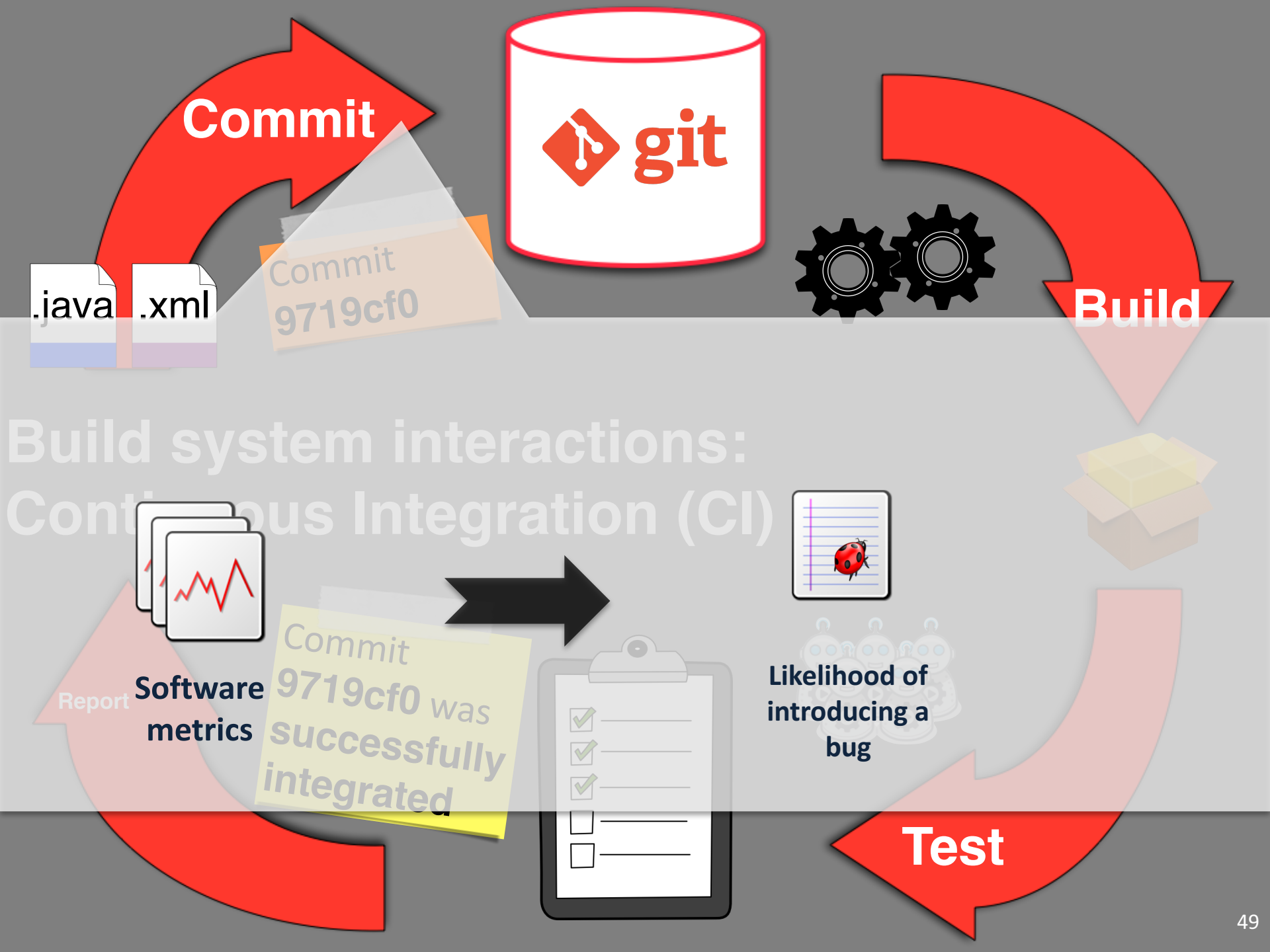
B. Ten *largest* functions

C. Ten *most-fixed* functions

Who produces more buggy code?

A. Junior Developer

B. Senior Developer


Commit
git
.java
.xml
Commit
9719cf0
Build
Build system interactions:
Continuous Integration (CI)
Report
Software
metrics
Commit
9719cf0 was
successfully
integrated
Likelihood of
introducing a
bug
Test

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

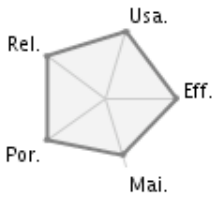
Comments
45,6%
15 789 lines
4 commented LOCs

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

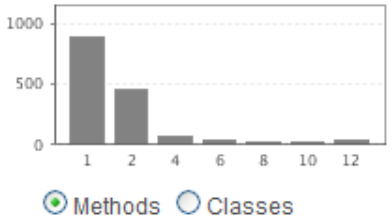
Rules compliance
79,0%



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



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

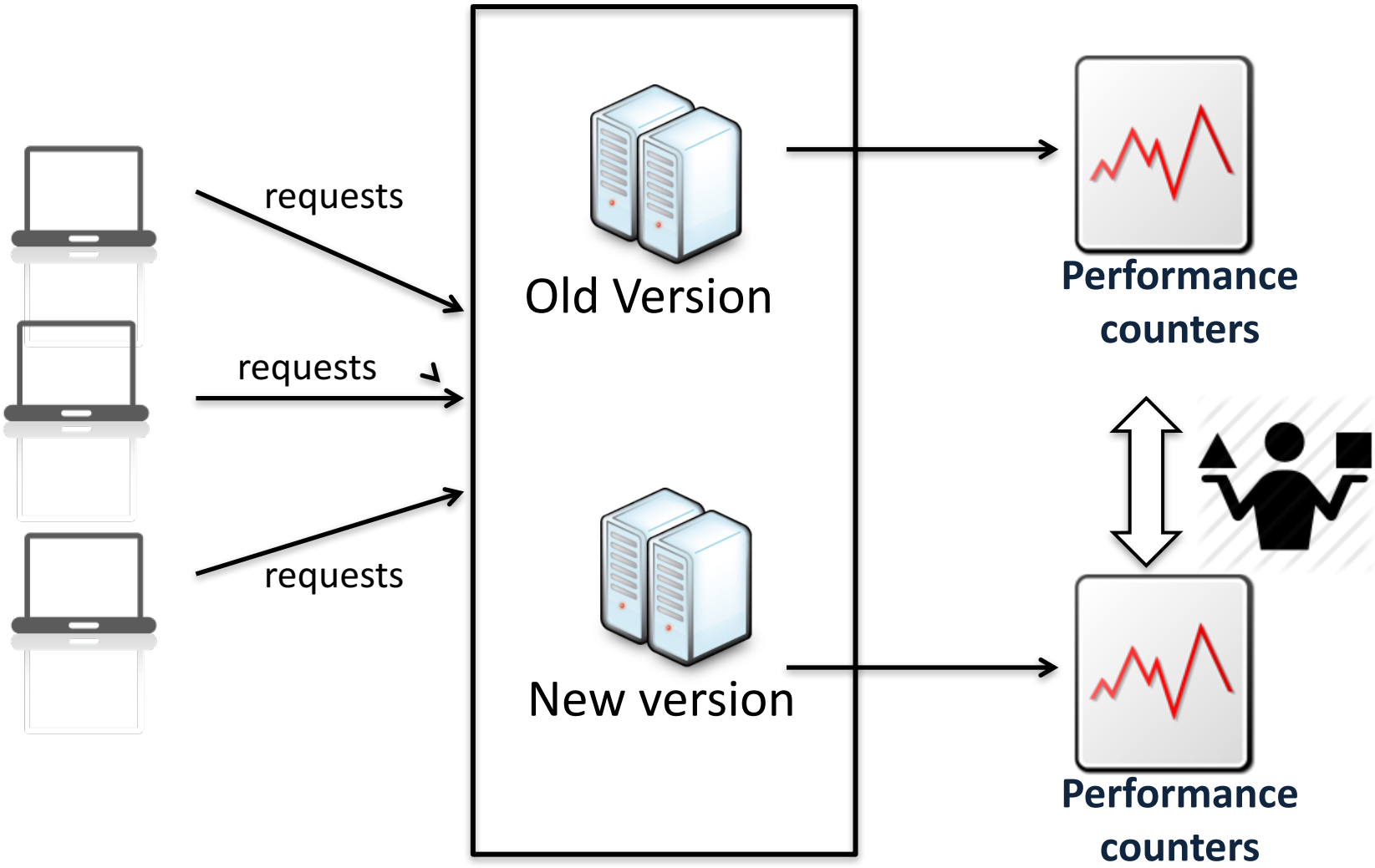
Detecting performance regression

What is a performance regression?

Old sion

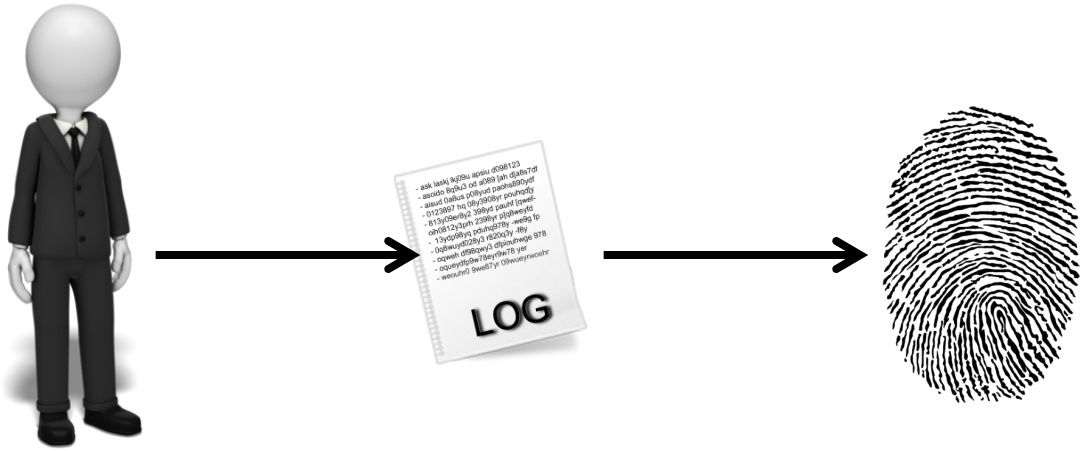
Does the new version have worse performance than the old version?

How to detect performance regression?



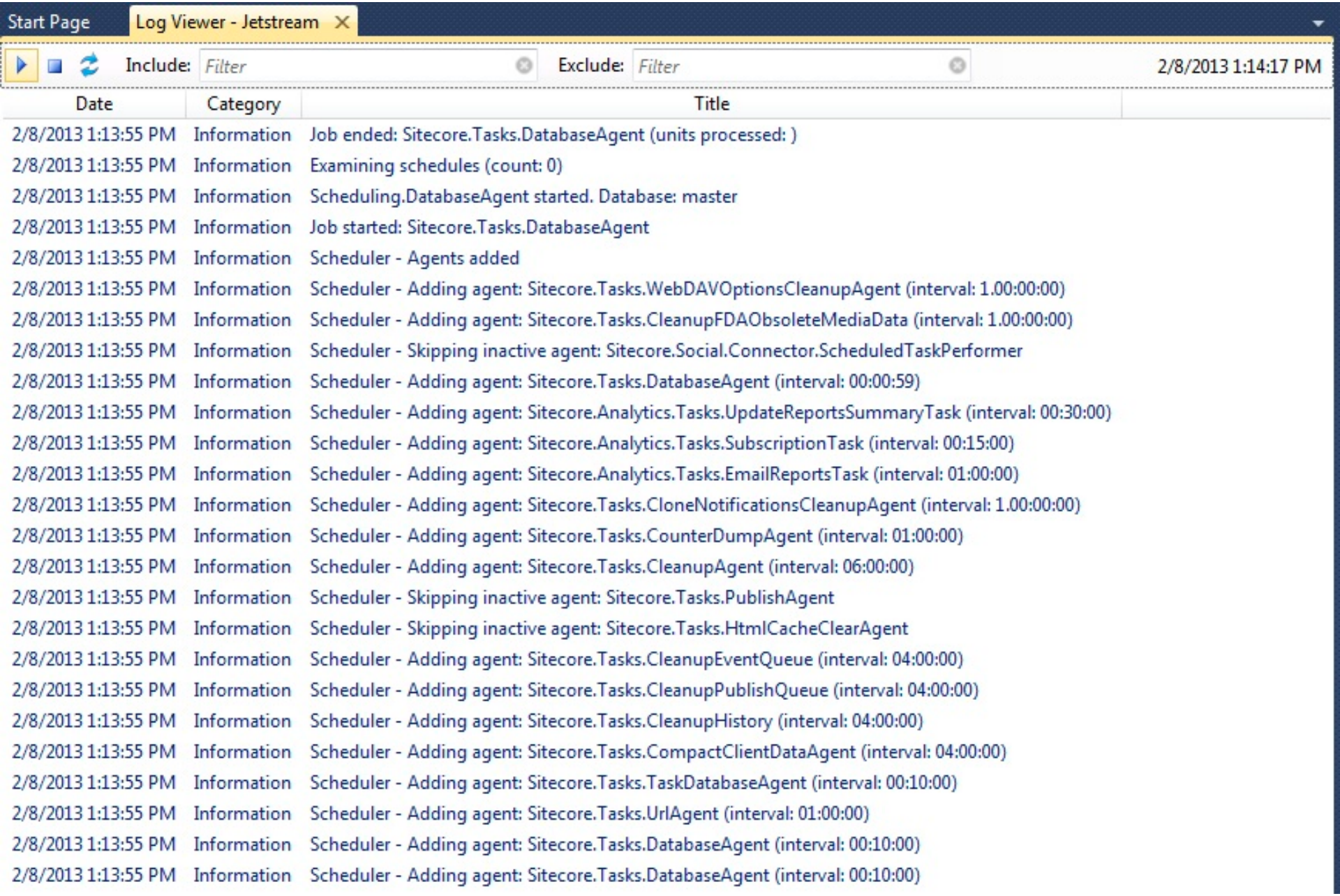
Are you testing realistically?

We can compare field and test workloads using logs



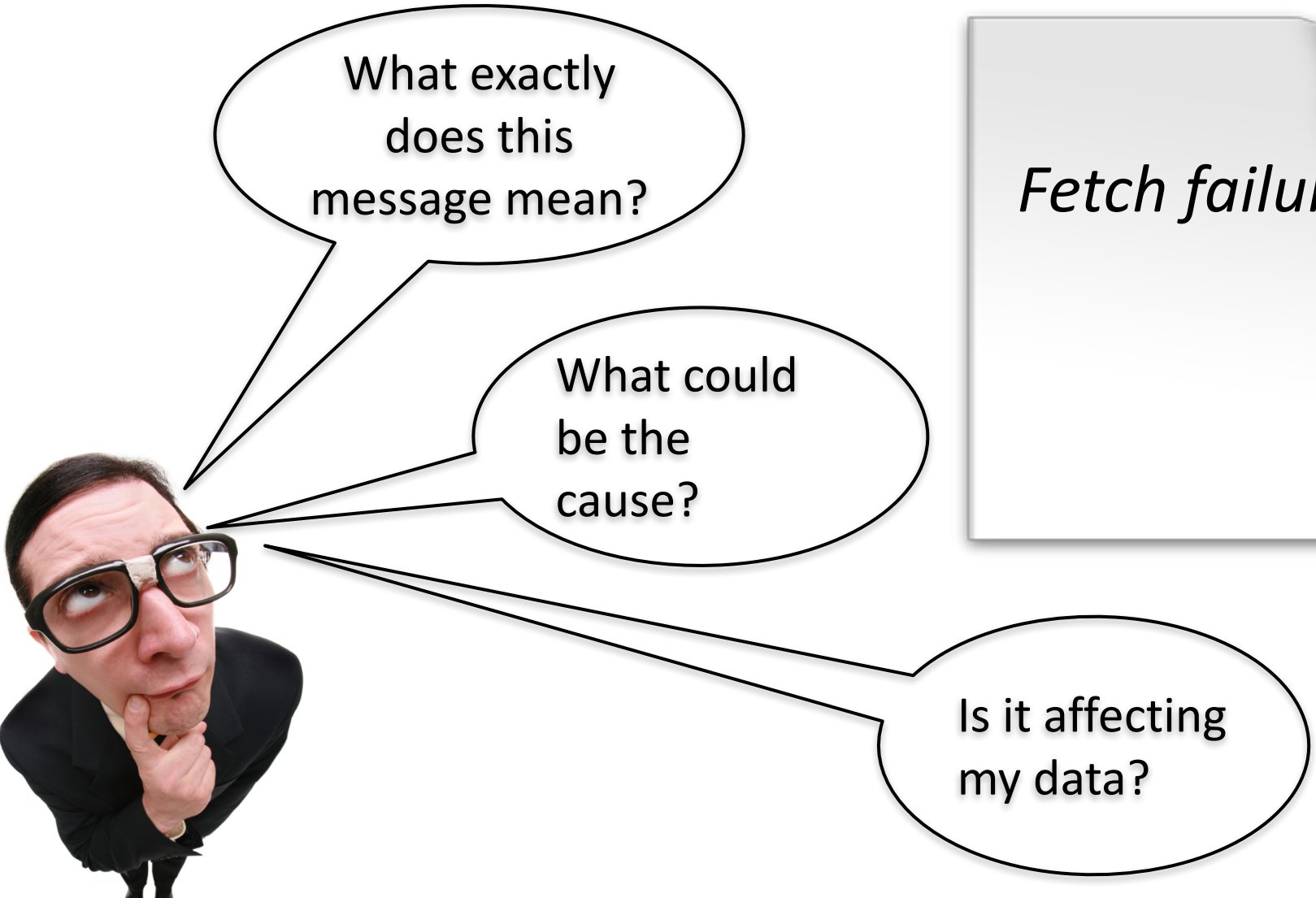
Is the behavior of this person covered in testing?

Understanding error messages



https://community.sitecore.net/technical_blogs/b/trevor_campbell/posts/28-days-of-sitecore-rocks-log-viewer

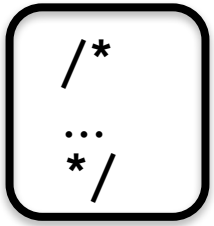
Practitioners have challenges in understanding log lines



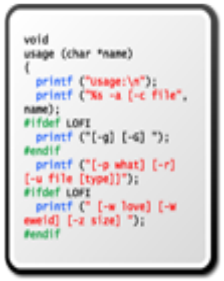
Attach development knowledge to logs

Source code

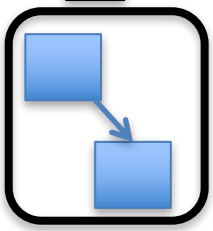
Issue reports



Code comments



Code commit



Call graph

How can these data help?

More will be covered in the class later.