An Exploration of the Challenges Associated with Software Logging in Large Systems

By: **Suhas Kabinna** (M.Sc. Thesis)
Supervisor: **Dr. Ahmed E. Hassan**
Date: May 27, 2016

Outline

- Background on logging statements
- Evolution of logging
- Challenges in modern day logging
- Logging library migrations
- Logging statement stability
- Conclusion

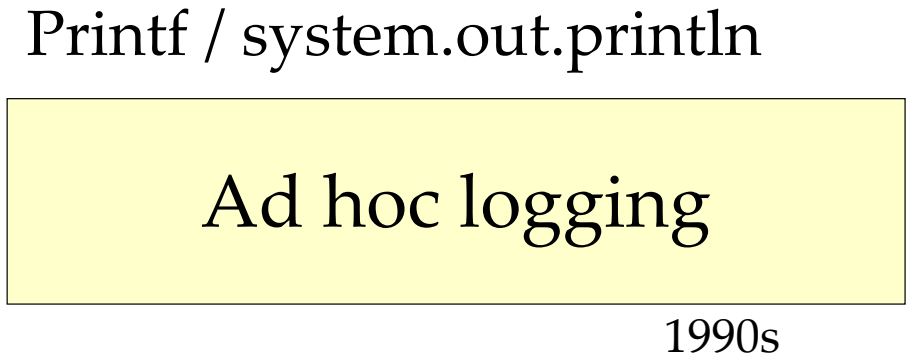
Logging statements

- Logging statements record useful information about the state of a system during its execution

```
function Thesis(){
      log.info("In Thesis");
      //…
      log.info("Exiting Thesis");
}
```

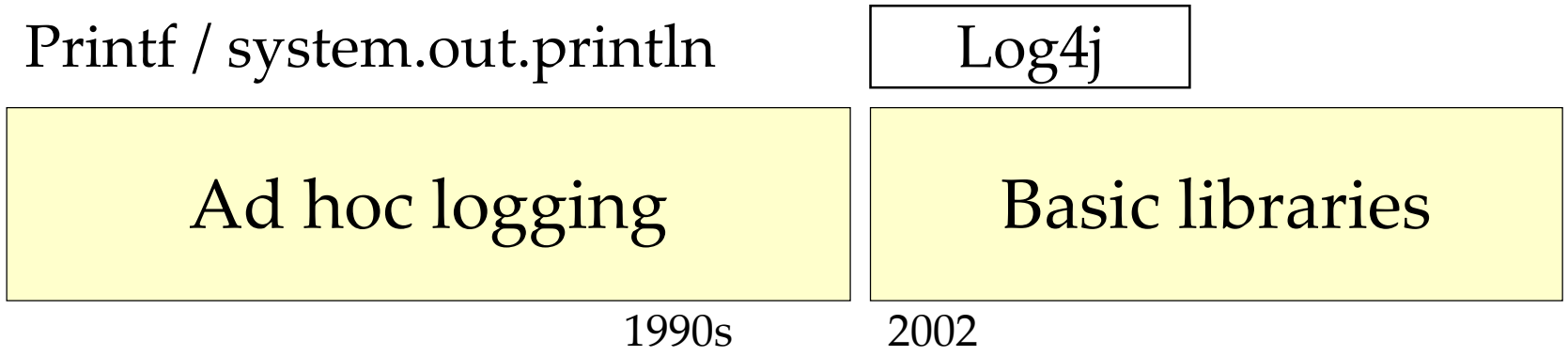
```
[13:40:00]  In Thesis
[13:40:05]  Exiting Thesis
```

Evolution of logging



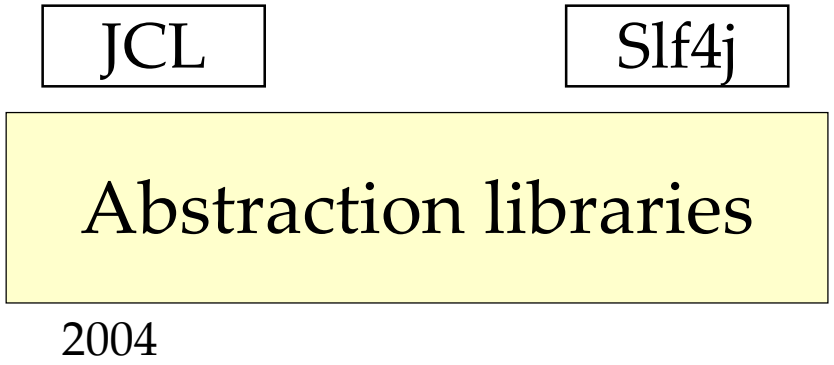
- Logging has evolved from simple *printf* statements into more complex and commonly used logging libraries

Evolution of logging



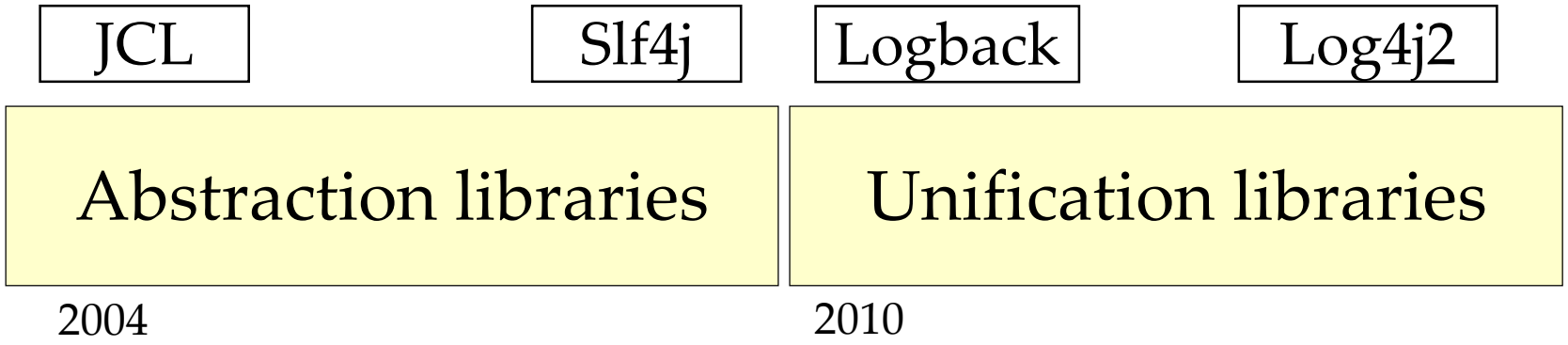
- Logging has evolved from simple *printf* statements into more complex and commonly used logging libraries
- Basic libraries provide logging levels and centralized output locations

Abstraction and unification libraries



- Abstraction libraries allow several different logging libraries to be used within the same project

Abstraction and unification libraries



- Abstraction libraries allow several different logging libraries to be used within the same project
- Unification libraries provide features of both basic and abstraction libraries

Challenges in modern day logging

Infrastructural Challenges

Processing Challenges

Part 1: Infrastructural challenges

Logging Library Migrations: A Case Study for the Apache Software Foundation Projects

Suhas Kabinna[1], Cor-Paul Bezemer[1], Weiyi Shang[2], Ahmed E. Hassan[1]
Software Analysis and Intelligence Lab (SAIL), Queen's University, Kingston, Canada[1]
Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada[2],
{kabinna, bezemer, ahmed}@cs.queensu.ca[1], shang@encs.concordia.ca[2]

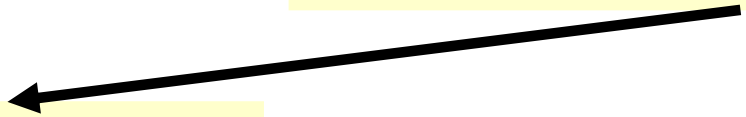
Published in MSR 2016

Infrastructural challenges

Developers migrate from one logging library to another

Infrastructural challenges

Developers migrate from one logging library to another



How often is the migration?

Approach


Projects
Keyword search
Identify logging library migrations

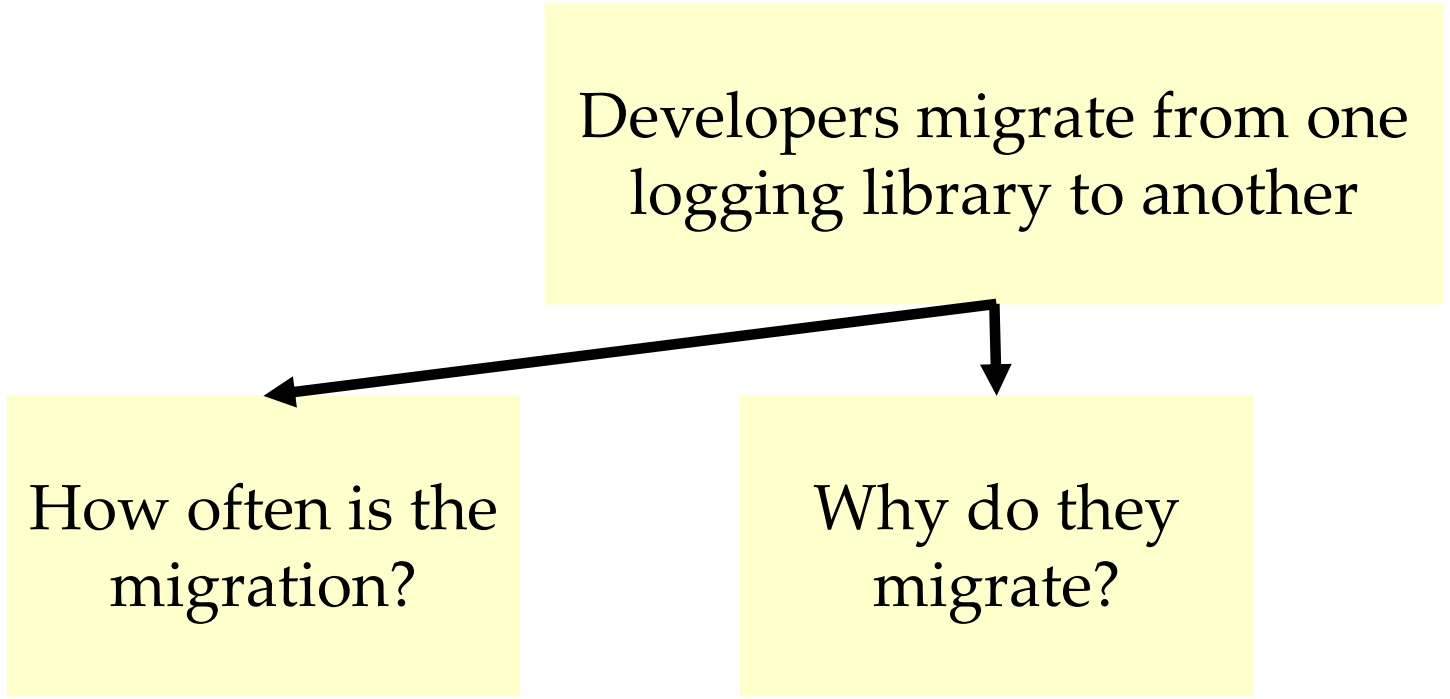
How often is the migration?

- 49 attempts identified in JIRA
- 33 projects successfully migrated (at least once)

14 migration attempts were abandoned

1. Nobody stepped up to make the required changes
2. Developers did not agree on the value of the migration

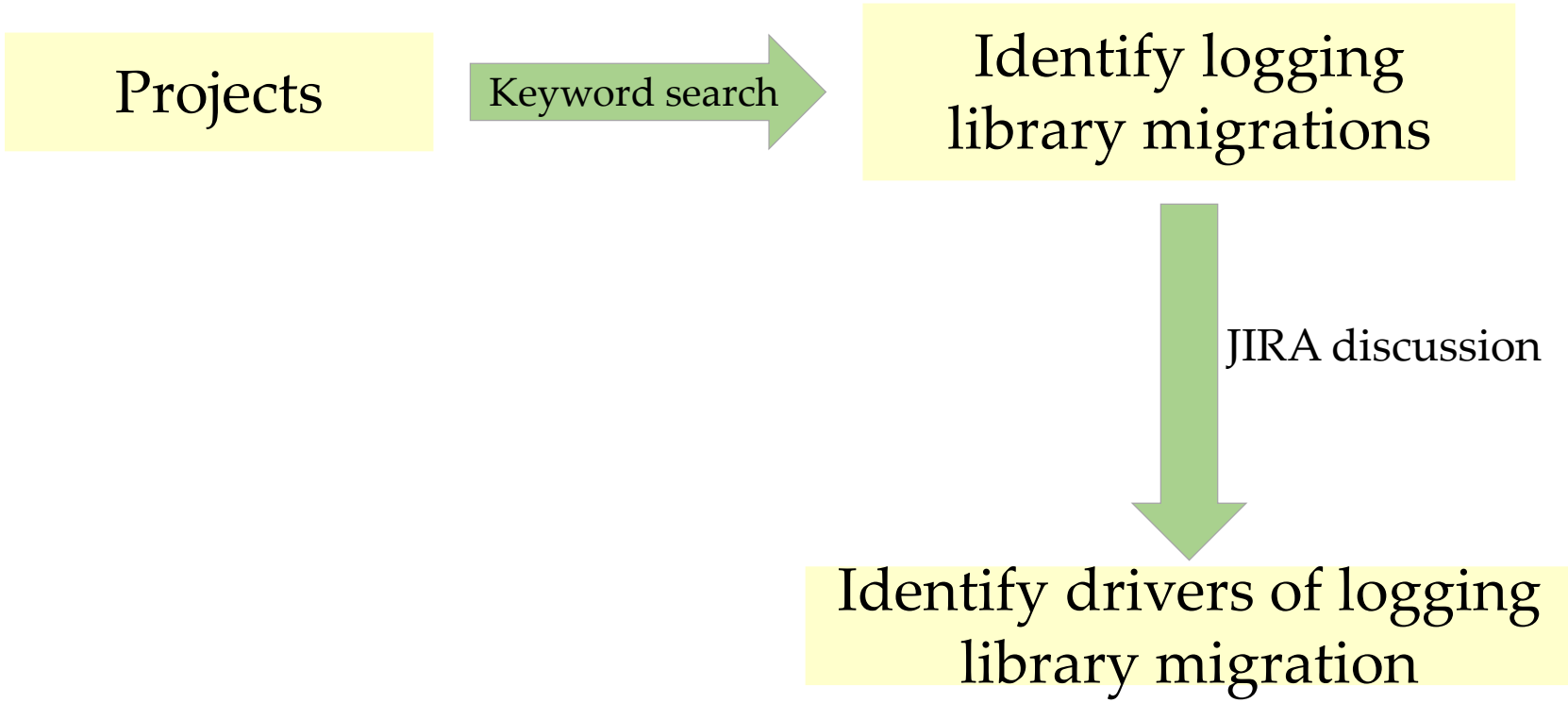
Infrastructural challenges



Approach


Projects
Keyword search
Identify logging library migrations

Approach


Projects
Keyword search
Identify logging library migrations
JIRA discussion
Identify drivers of logging library migration

Why do developer migrate logging libraries?

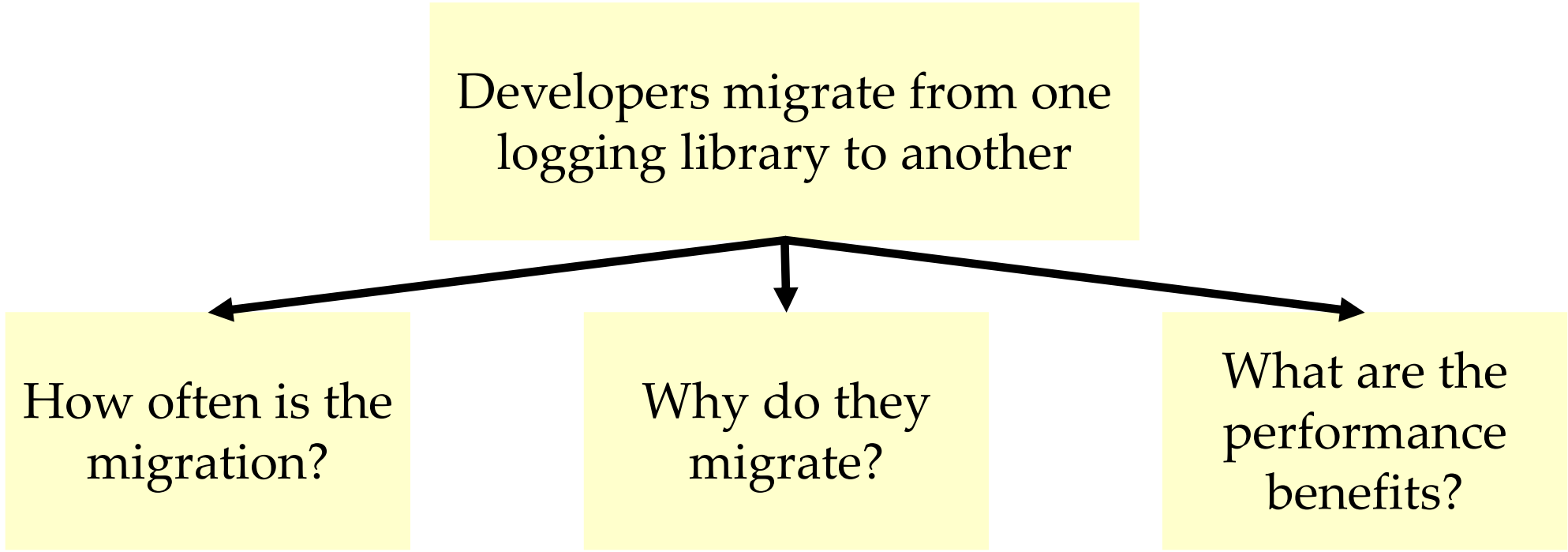
1	Flexibility	57.4%
2	Performance	37.0%
3	Code maintenance	33.3%

Based on manual tagging of JIRA issues that are related to logging library migration (multiple tags per issue possible)

Migrations can lead to problems

24 out of 33 projects encounter an average of 2 post-migration bugs due to the migration

Infrastructural challenges

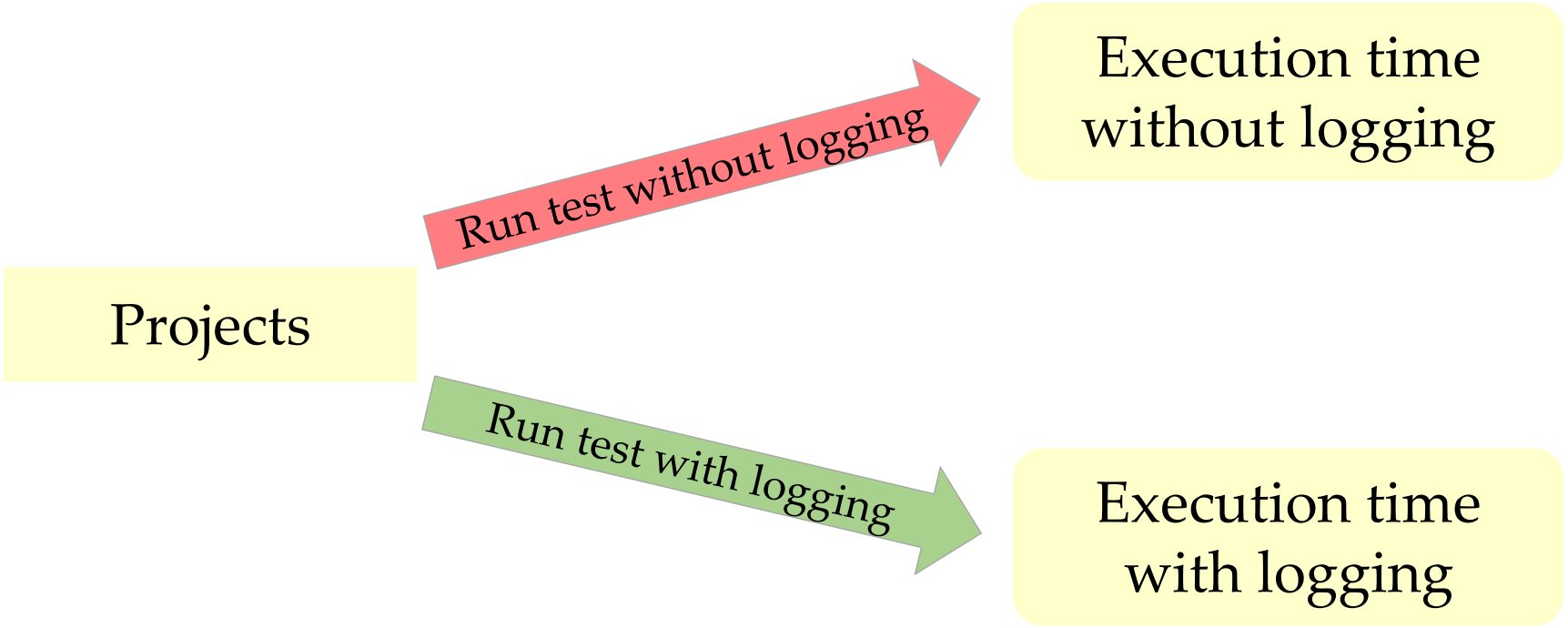


Was the migration justified?

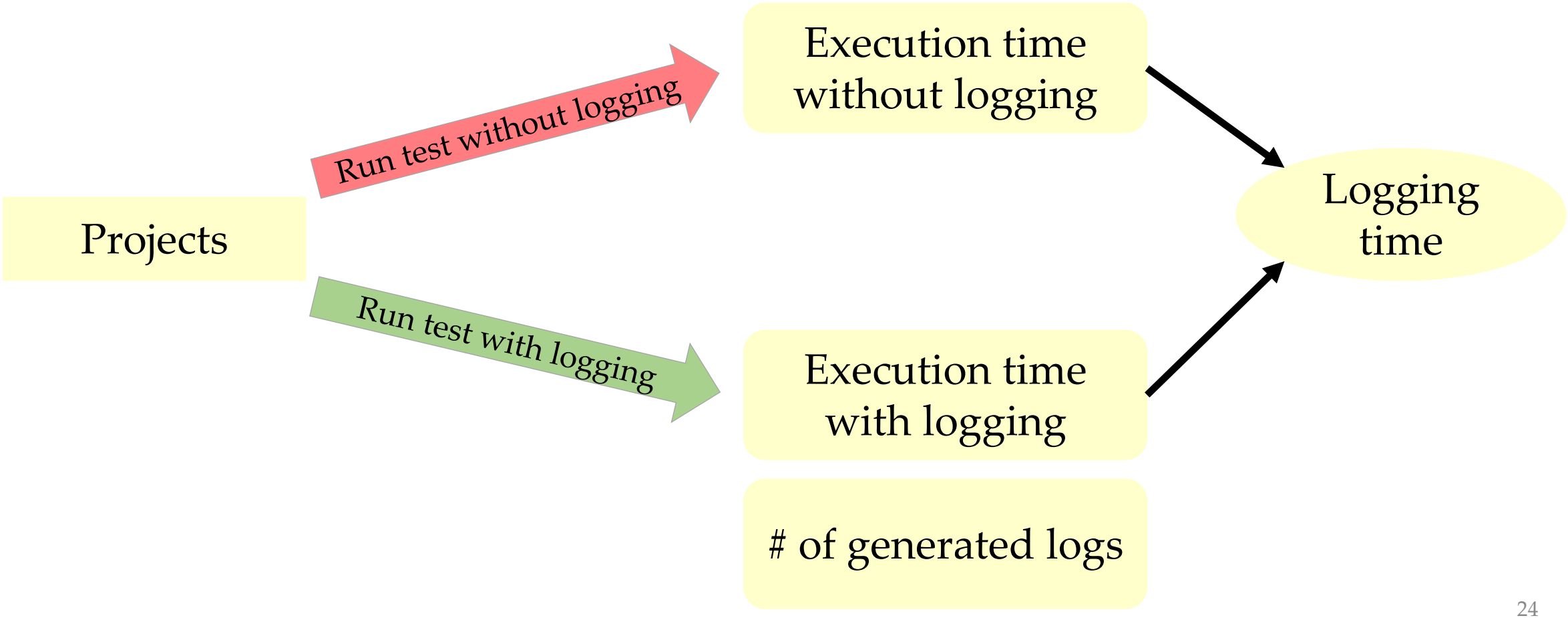
We calculate the time to generate output of one logging statement pre and post-migration

What are the performance benefits?

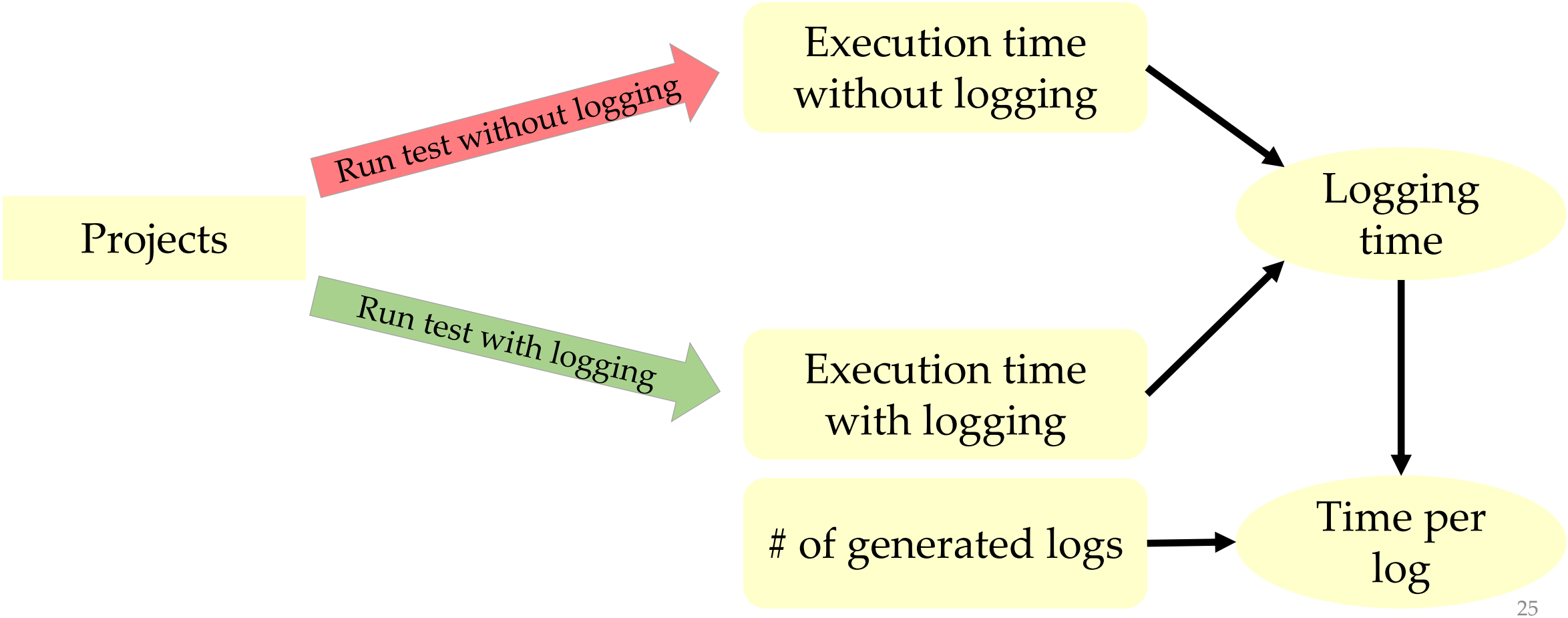
What are the performance benefits?



What are the performance benefits?



What are the performance benefits?



There is a 28 - 44% speedup of logging after migration

There is a 28 - 44% speedup of logging after migration

However…

in 2 out of 3 projects this speedup is only noticeable at debug level

(so NOT in practice!)

Infrastructural challenges conclusion

Developers should better estimate the needed effort, the performance improvements achieved from migration and plan for avoiding post-migration bugs

Part 2: Processing challenges

Examining the Stability of Logging Statements

Suhas Kabinna[1], Weiyi Shang[2], Cor-Paul Bezemer[1],Ahmed E. Hassan[1]
Software Analysis and Intelligence Lab (SAIL),Queen's University,Kingston, Ontario [1]
Department of Computer Science and Software Engineering Concordia University,Montreal, QC, Canada[2],
Email:{kabinna, bezemer, ahmed}@cs.queensu.ca[1]
shang@encs.concordia.ca[2]

Published in SANER 2016
Extension – EMSE (under review)

Terabytes of logs generated every hour !!!

Processing challenges

Release 1

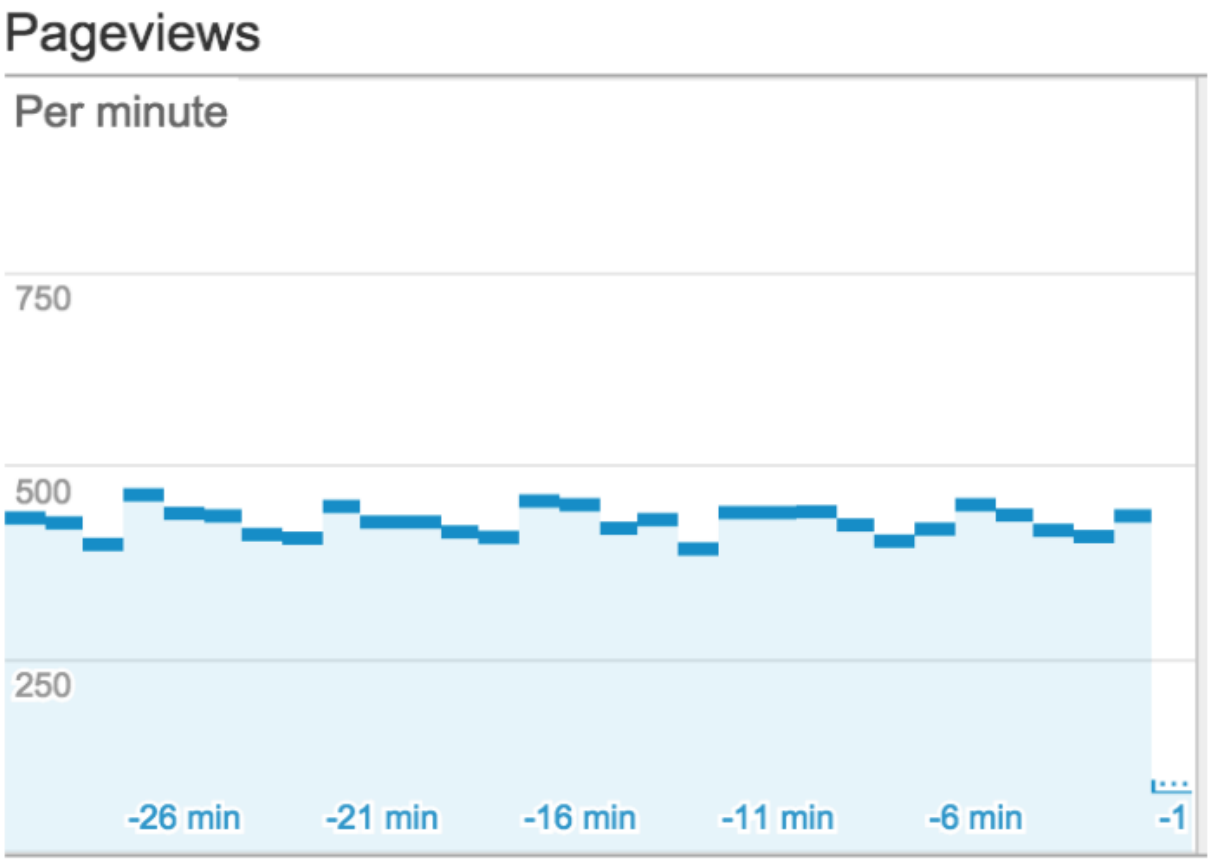
```
..
Log.info("Connection request received from {}",
ip_address);
..
Log.info("Connecting to {}", Ip_address);
```

Processing challenges

Right now

1946

active users on site

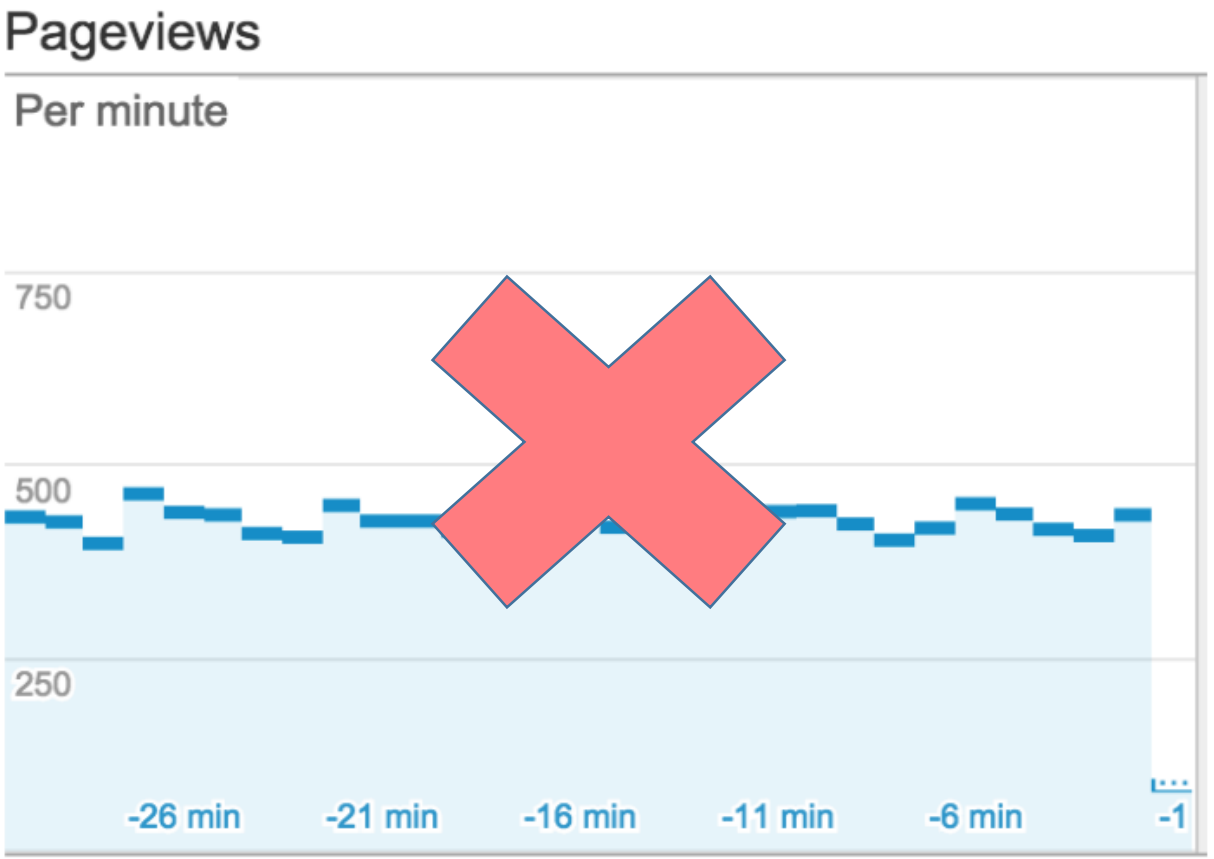


Processing challenges

Release 2

```
..
Log.info("Connection request received from {}",
ip_address);
..
Log.info(" New Connection");
```

Processing challenges

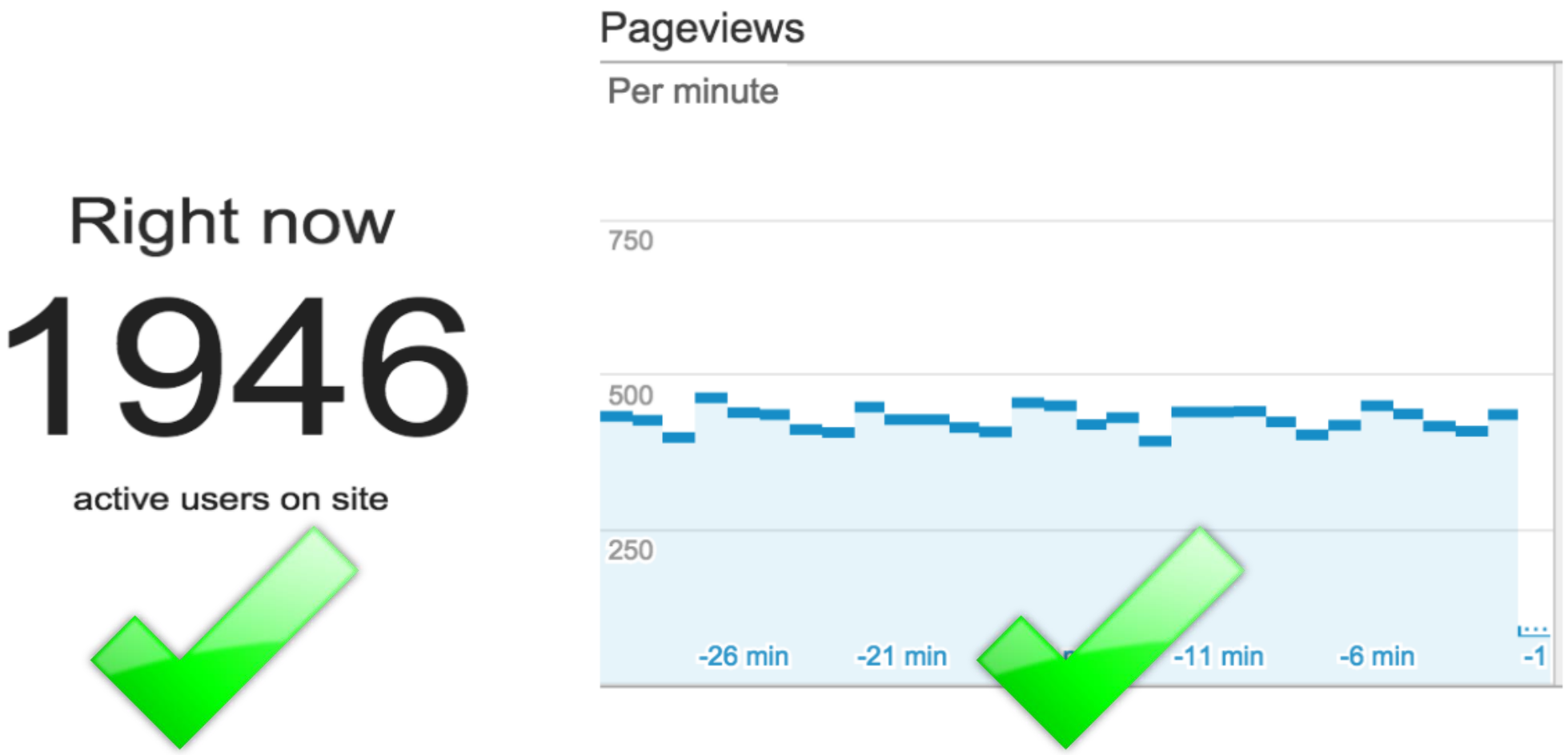


Processing challenges

Release 2

```
..
Log.info("Connection request received from {}",
ip_address);
..
Log.info(" New Connection");
```

Processing challenges

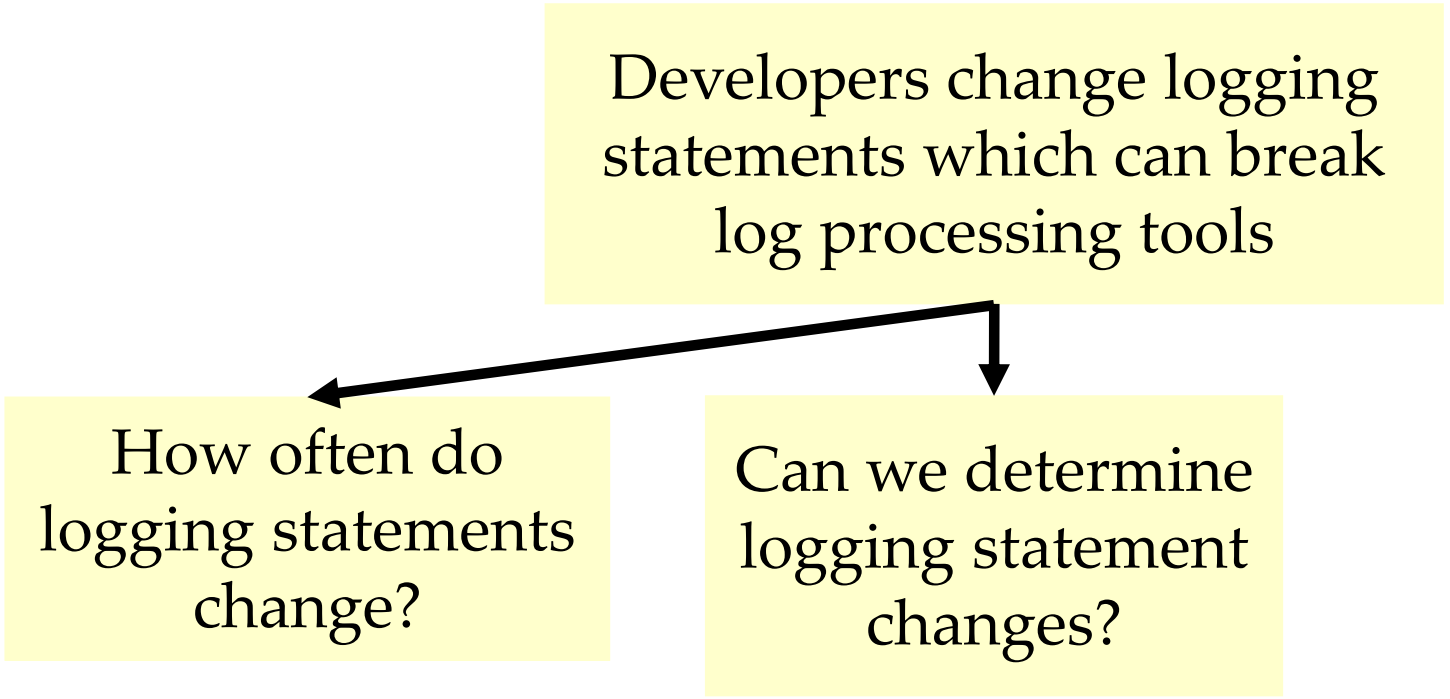


Processing challenges

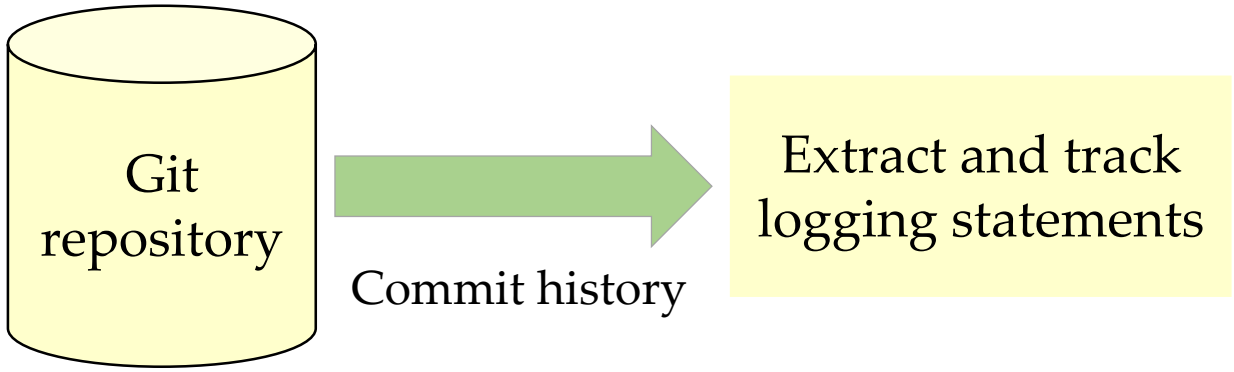
Developers change logging statements which can break log processing tools

How often do logging statements change?

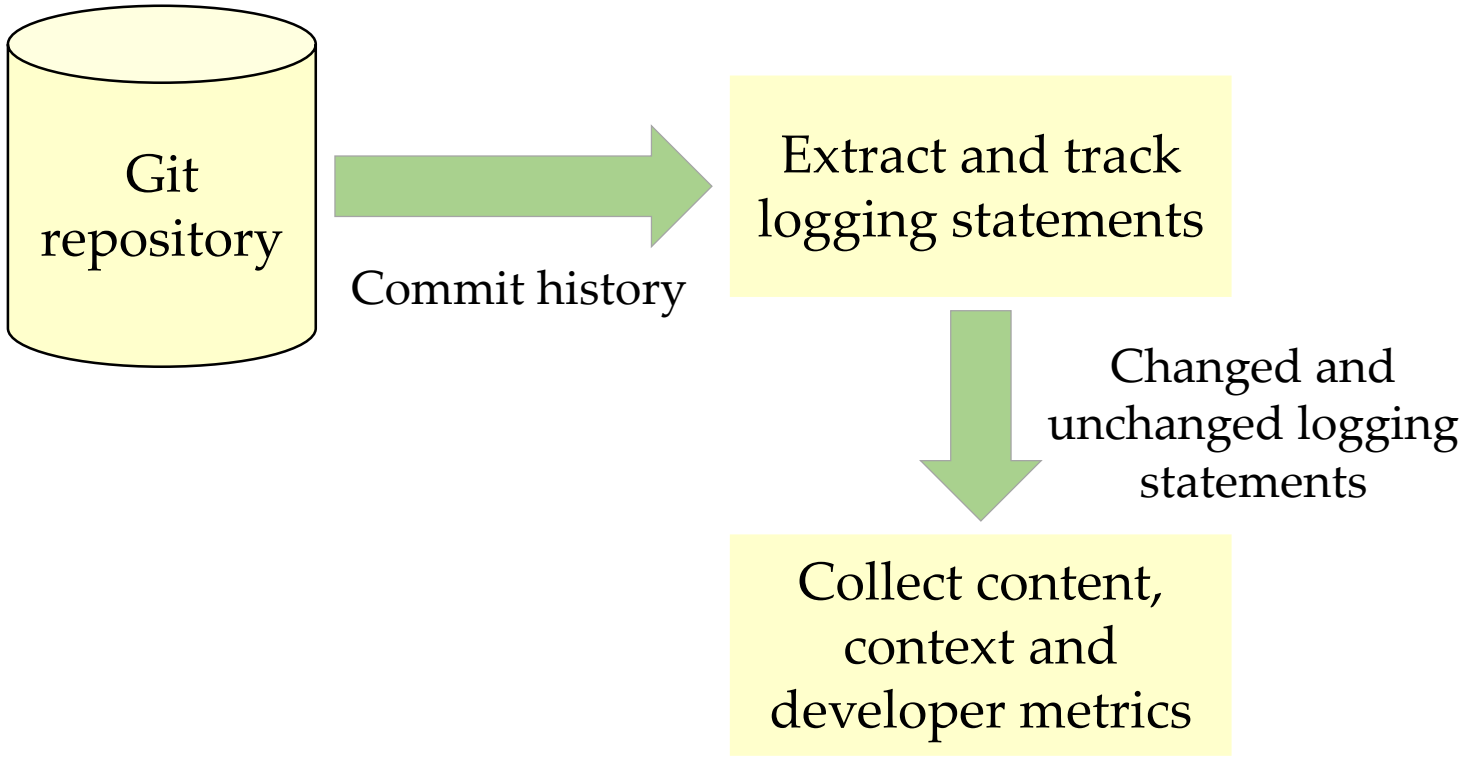
Processing challenges



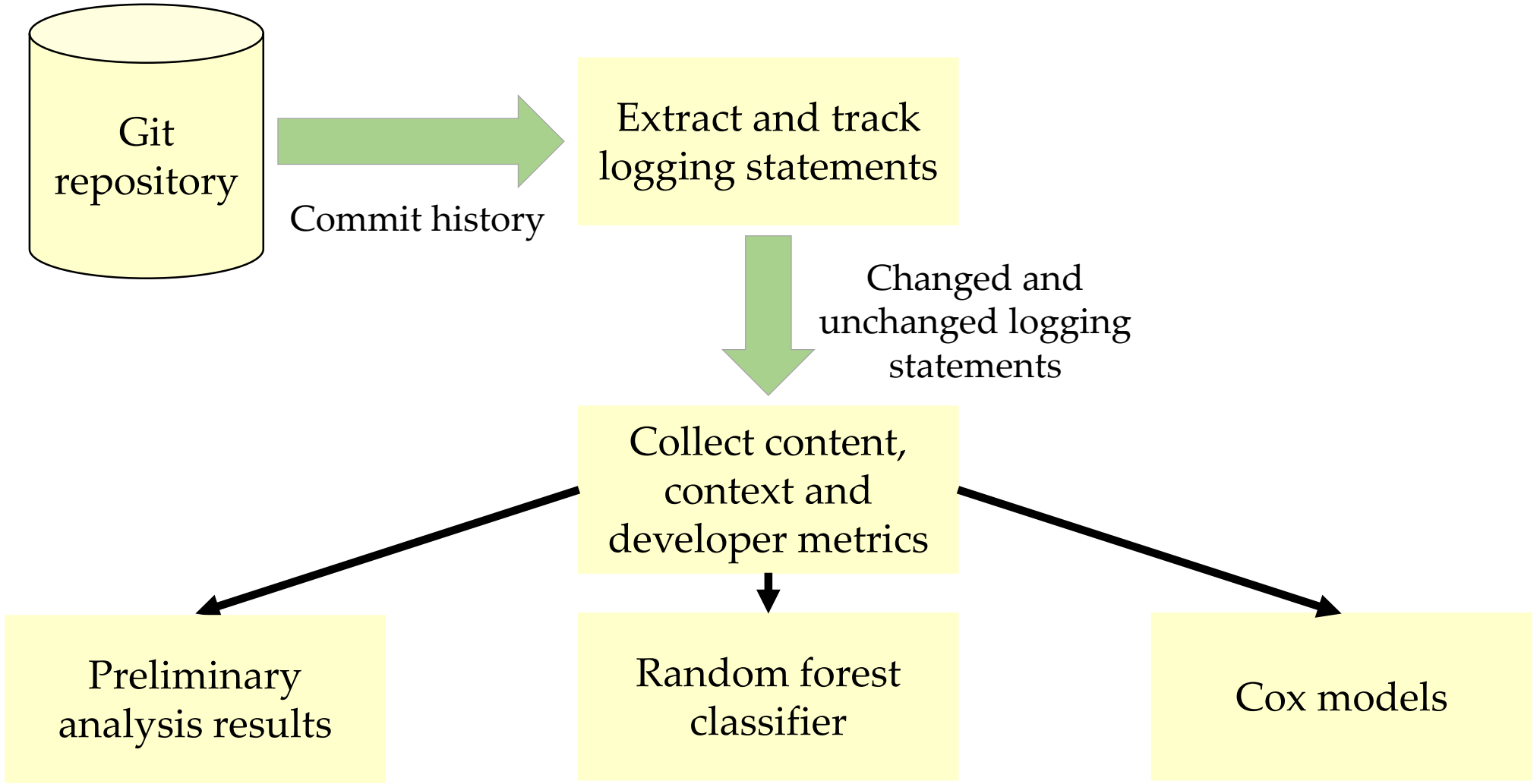
Approach


Git repository
Commit history
Extract and track logging statements

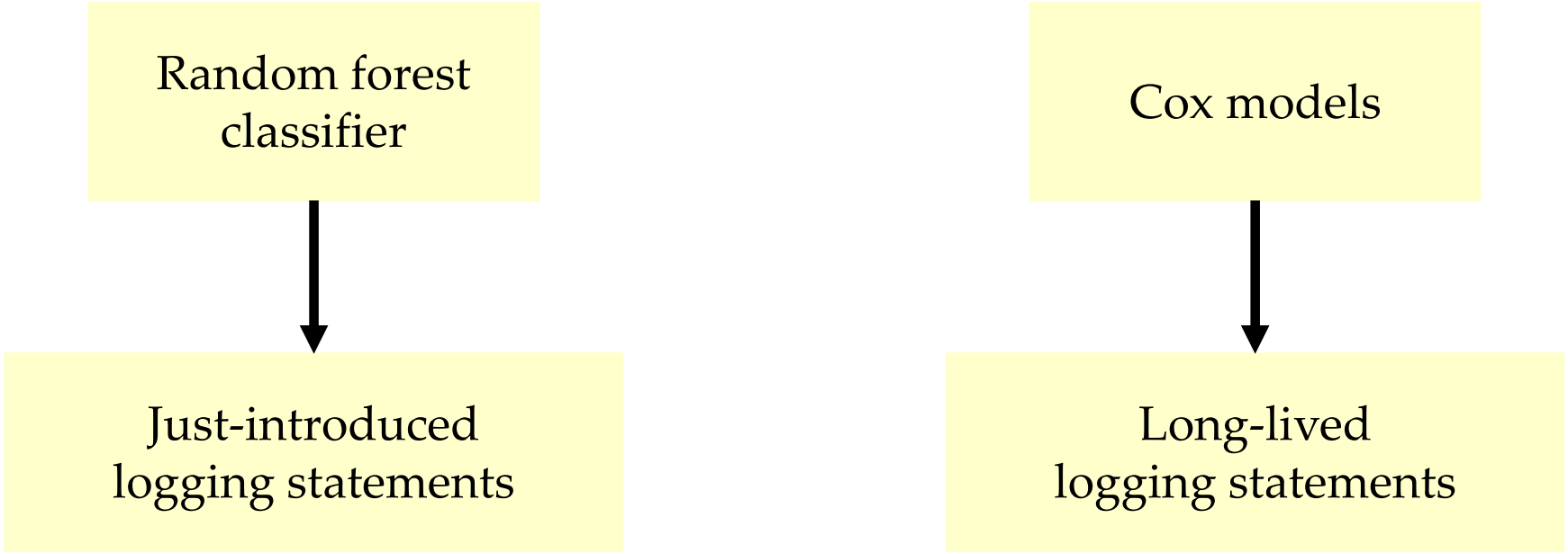
Approach


Git repository
Commit history
Extract and track logging statements
Changed and unchanged logging statements
Collect content, context and developer metrics

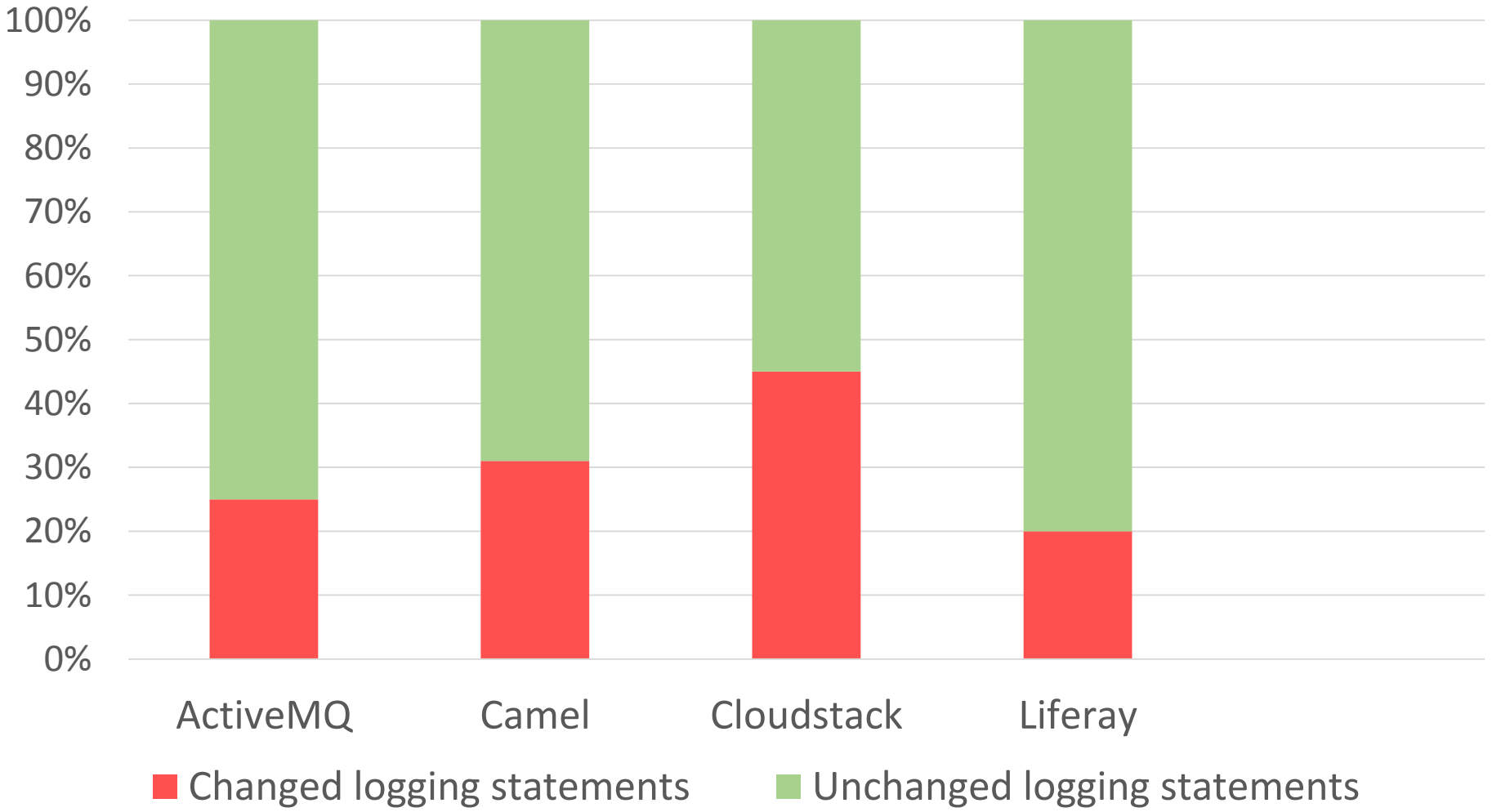
Approach


Git repository
Commit history
Extract and track logging statements
Changed and unchanged logging statements
Collect content, context and developer metrics
Preliminary analysis results
Random forest classifier
Cox models

Random forest classifiers and Cox models



20% - 45% of logging statements are changed at-least once

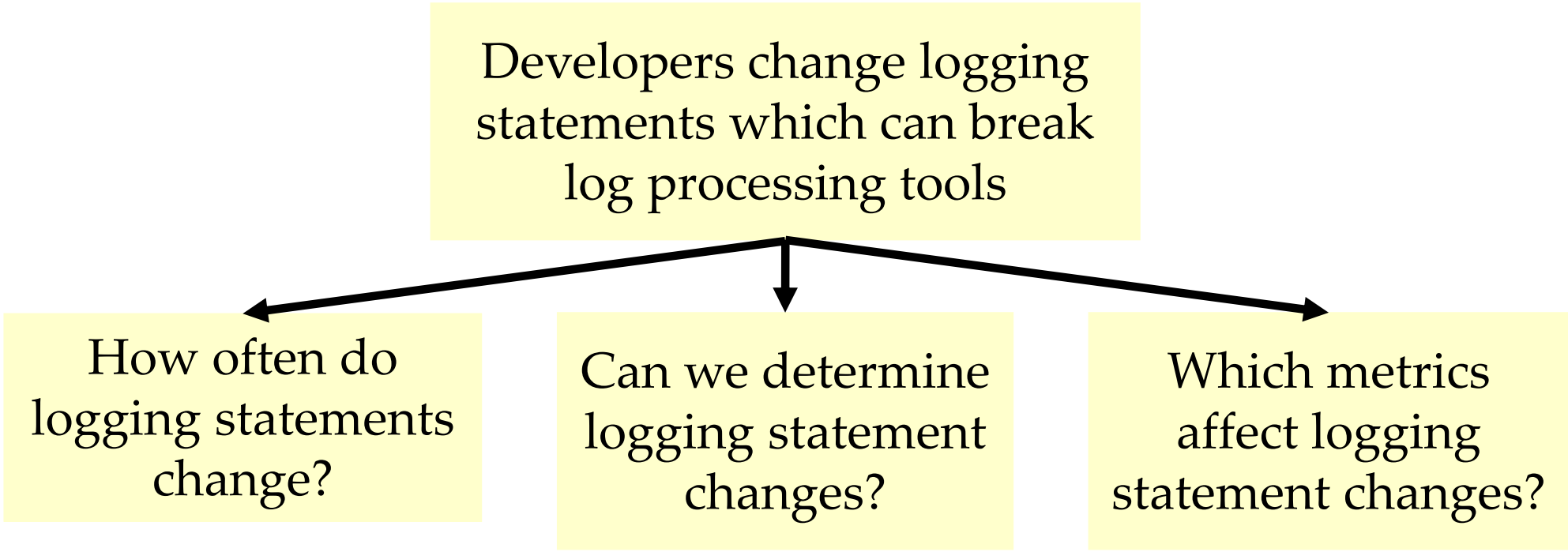


We can accurately determine the likelihood of just-introduced logging statements

- Our **random forest** classifiers achieve a precision of 0.83-0.91 and a recall of 0.65-0.85 for just-introduced logging statements

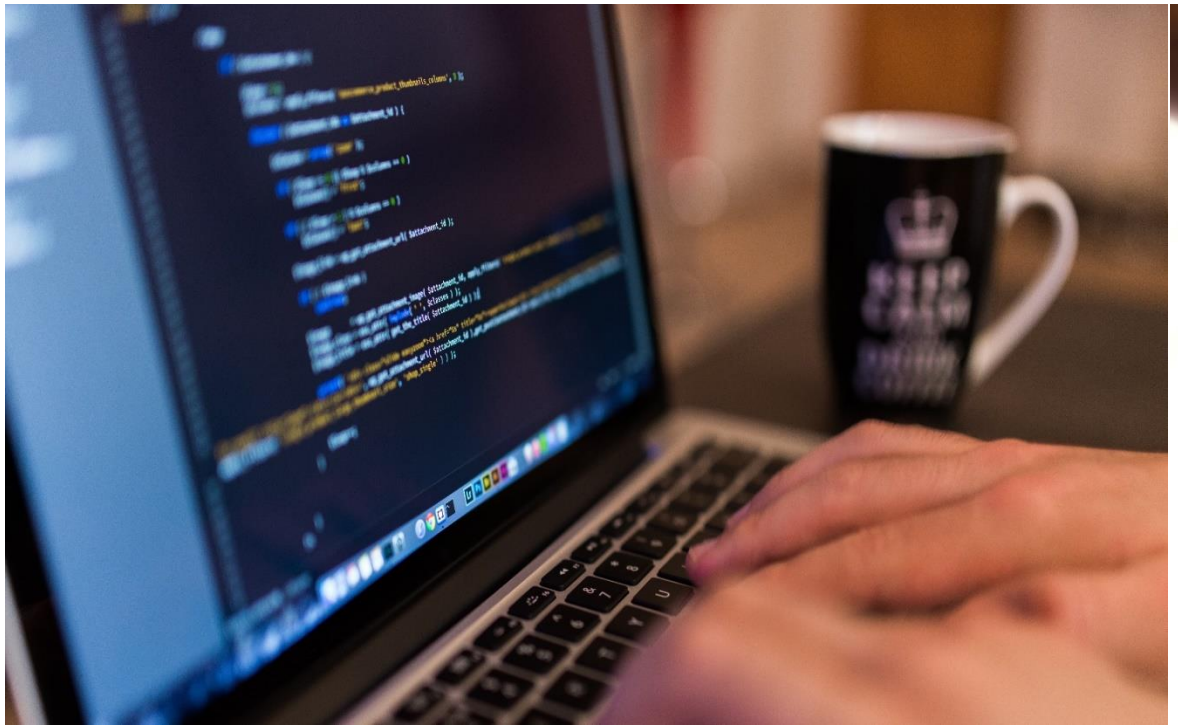
- Our **Cox models** are a good fit and can accurately determine the likelihood of long-lived logging statements changing

Processing challenges



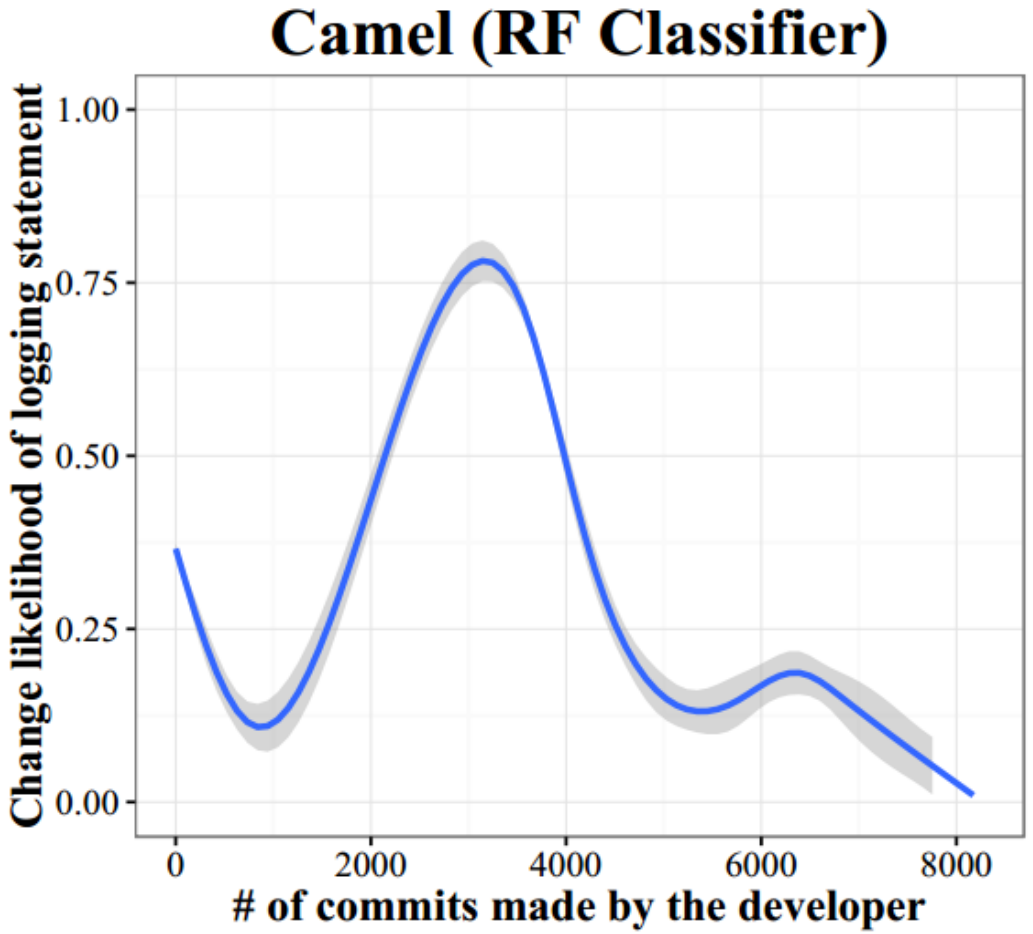
Important metric to determine the likelihood of logging statements

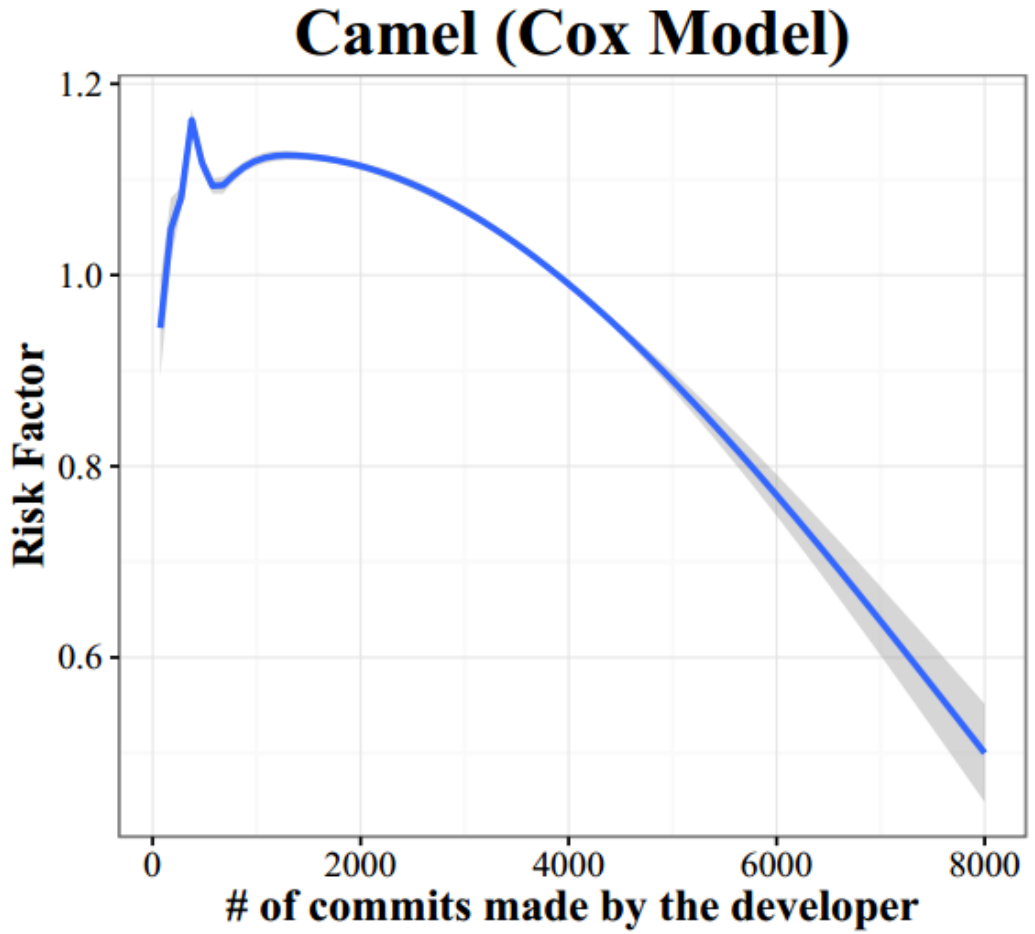
Developer experience



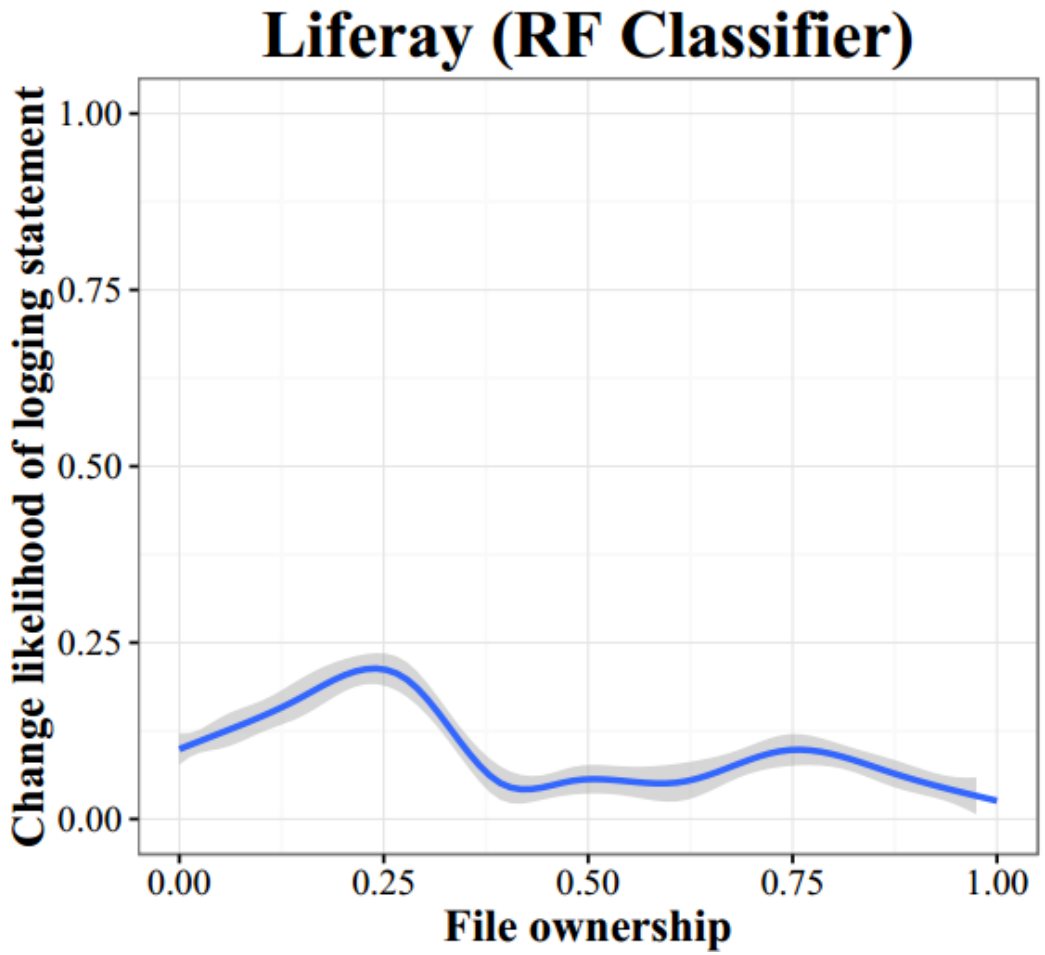
File ownership

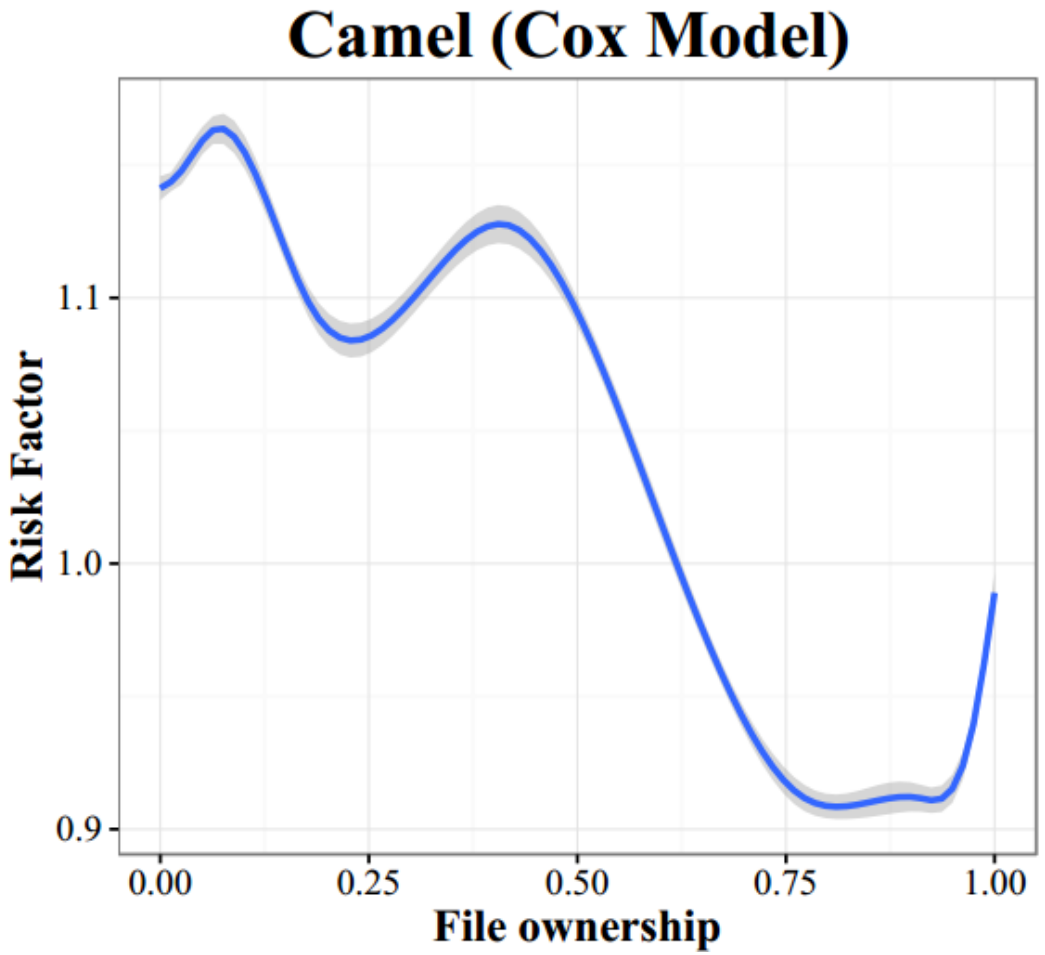
Developer experience is an important metric to determine the likelihood of logging statements



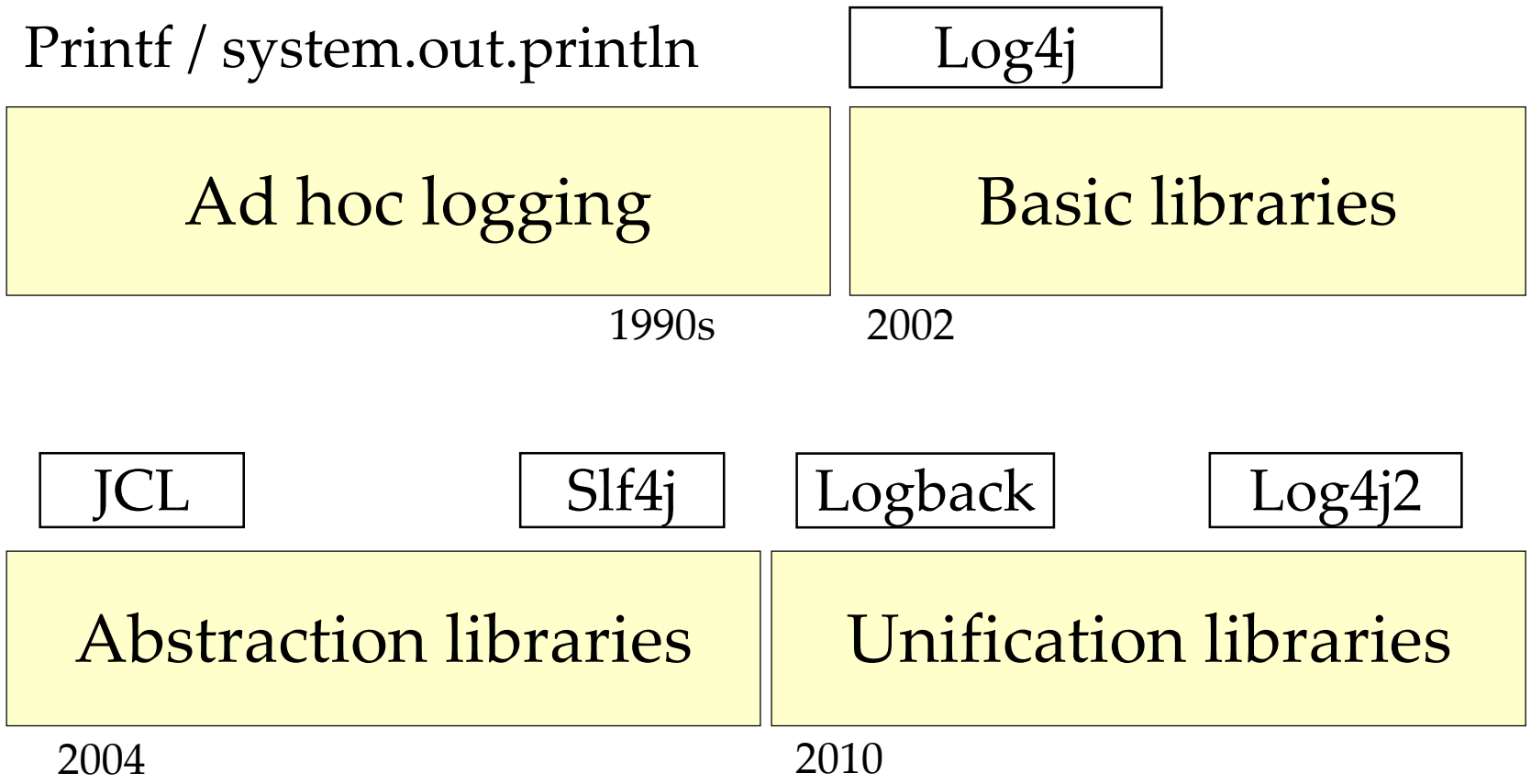


Logging statements by owners of the file are less likely to be changed

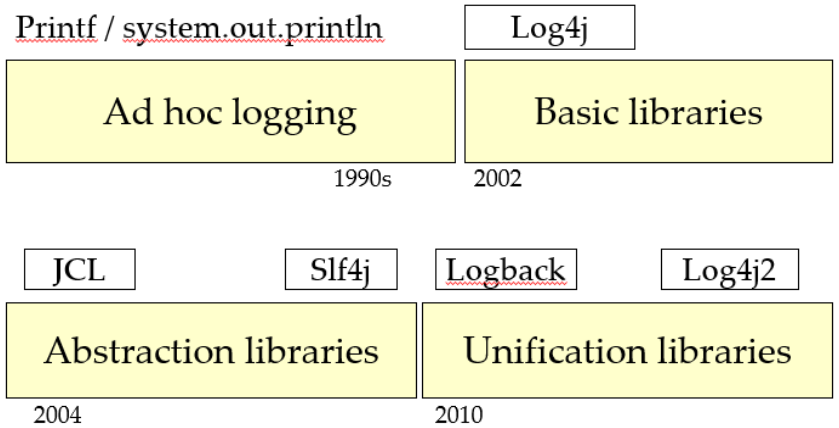




Evolution of logging



Evolution of logging

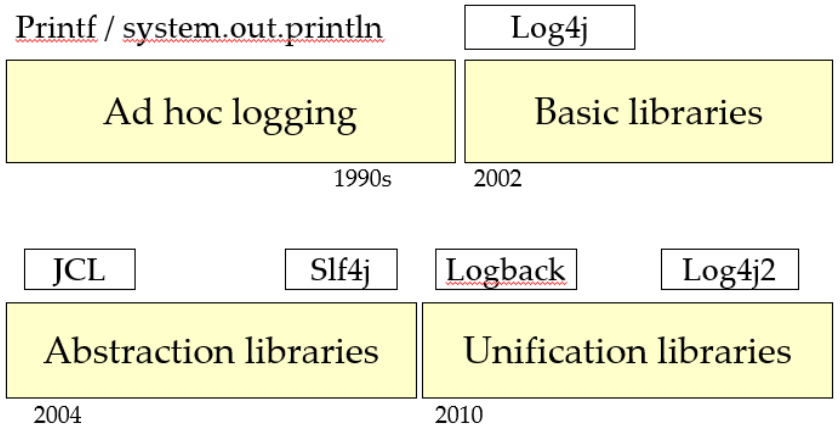


38

Infrastructural challenges Conclusion

Developers should better estimate the needed effort, the performance improvements achieved from migration and plan for avoiding post-migration bugs.

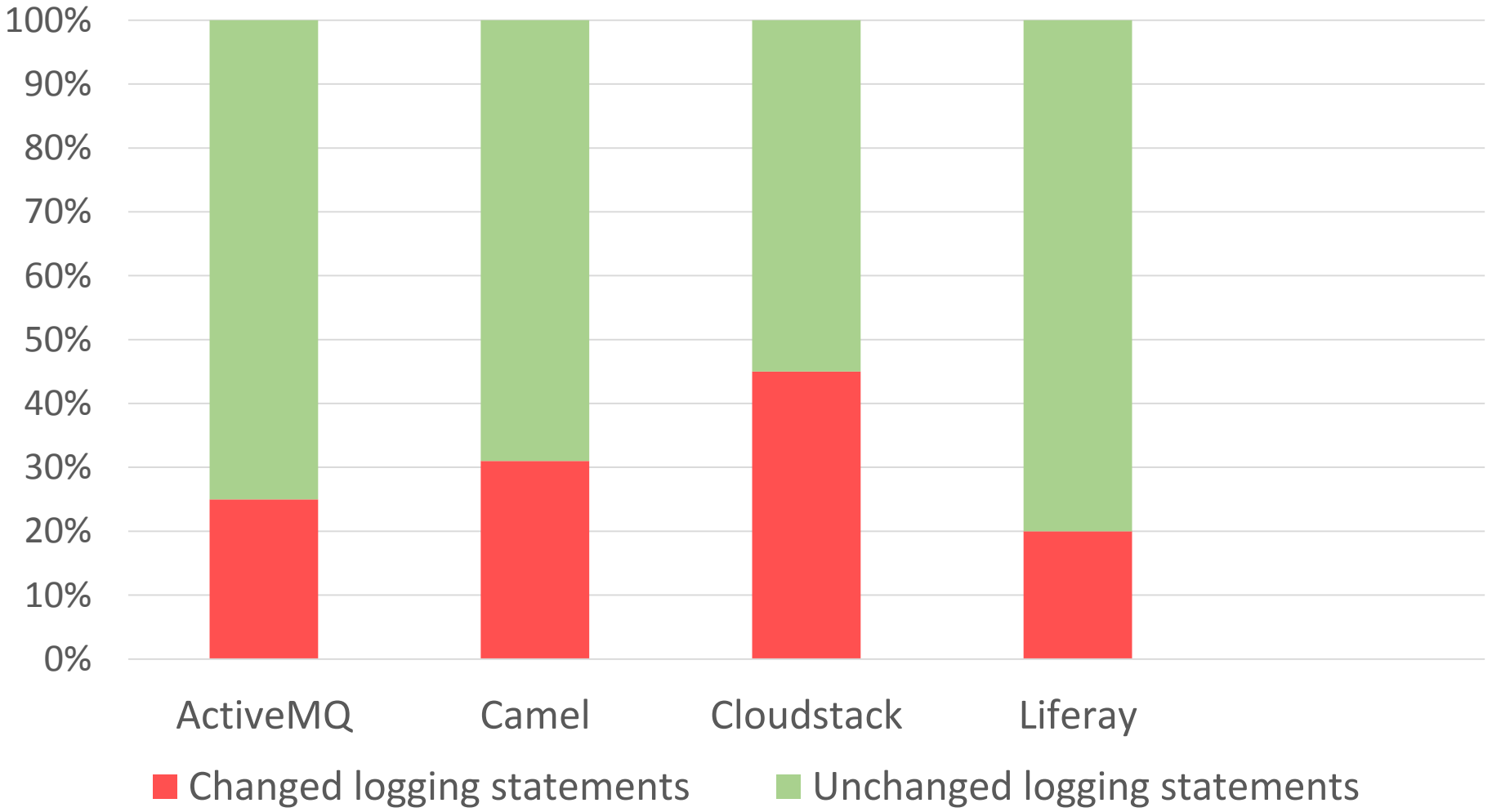
Evolution of logging



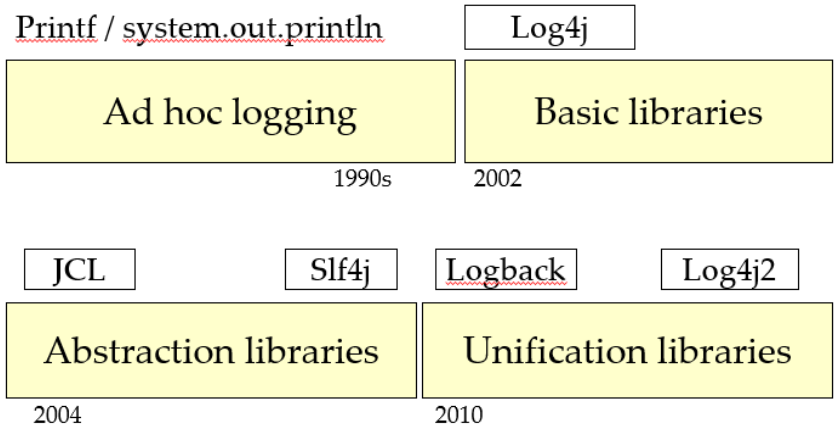
Infrastructural challenges Conclusion

Developers should better estimate the needed effort, the performance improvements achieved from migration and plan for avoiding post-migration bugs.

20% - 45% of logging statements are changed at-least once



Evolution of logging



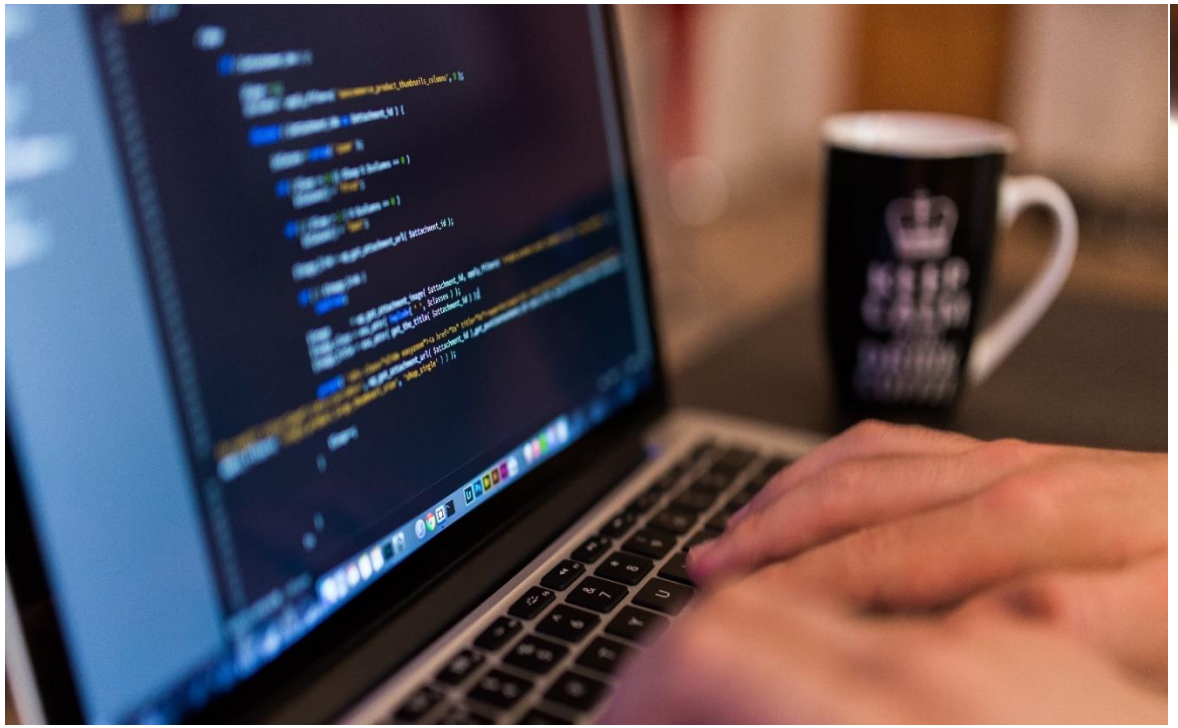
20% - 45% of logging statements are changed at-least once

Infrastructural challenges Conclusion

Developers should better estimate the needed effort, the performance improvements achieved from migration and plan for avoiding post-migration bugs.

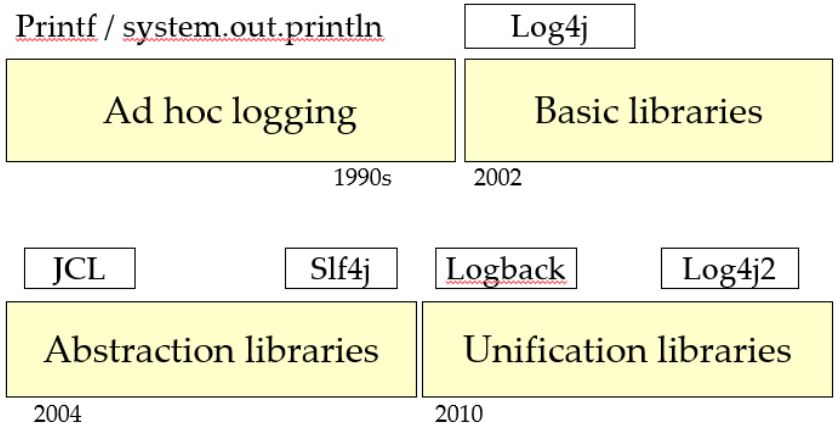
Important metric to determine the likelihood of logging statements

Developer experience



File ownership

Evolution of logging



Infrastructural challenges Conclusion

Developers should better estimate the needed effort, the performance improvements achieved from migration and plan for avoiding post-migration bugs.

20% - 45% of logging statements are changed at-least once

Important metric to determine the likelihood of logging statements

Developer experience

File ownership