Why do big data and cloud systems slow down and stop?

Shan Lu

What are?

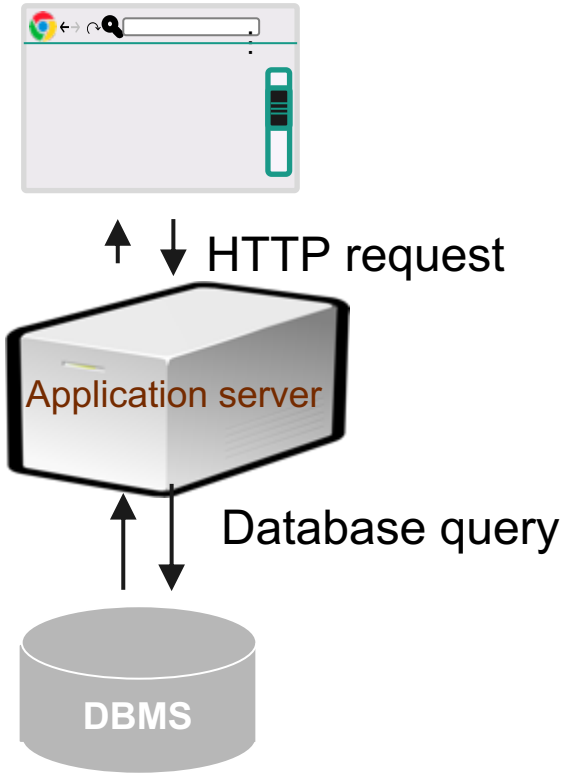
Why do big data and cloud systems slow down and stop?

Big data & cloud systems

Big data & cloud systems

- DB-backed web applications
- Cloud services

DB-backed web applications



Performance is critical for web applications

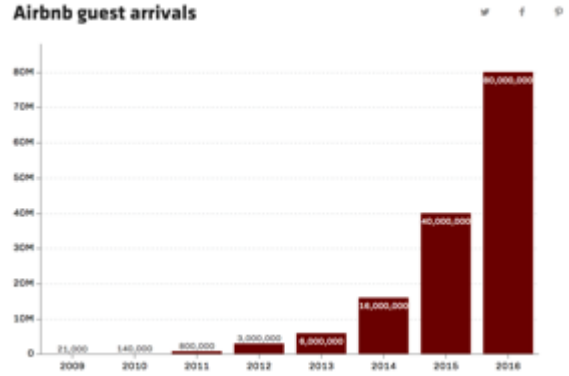
- Low latency is critical

Nearly **half** of the users expect a site to load in less than **2** seconds

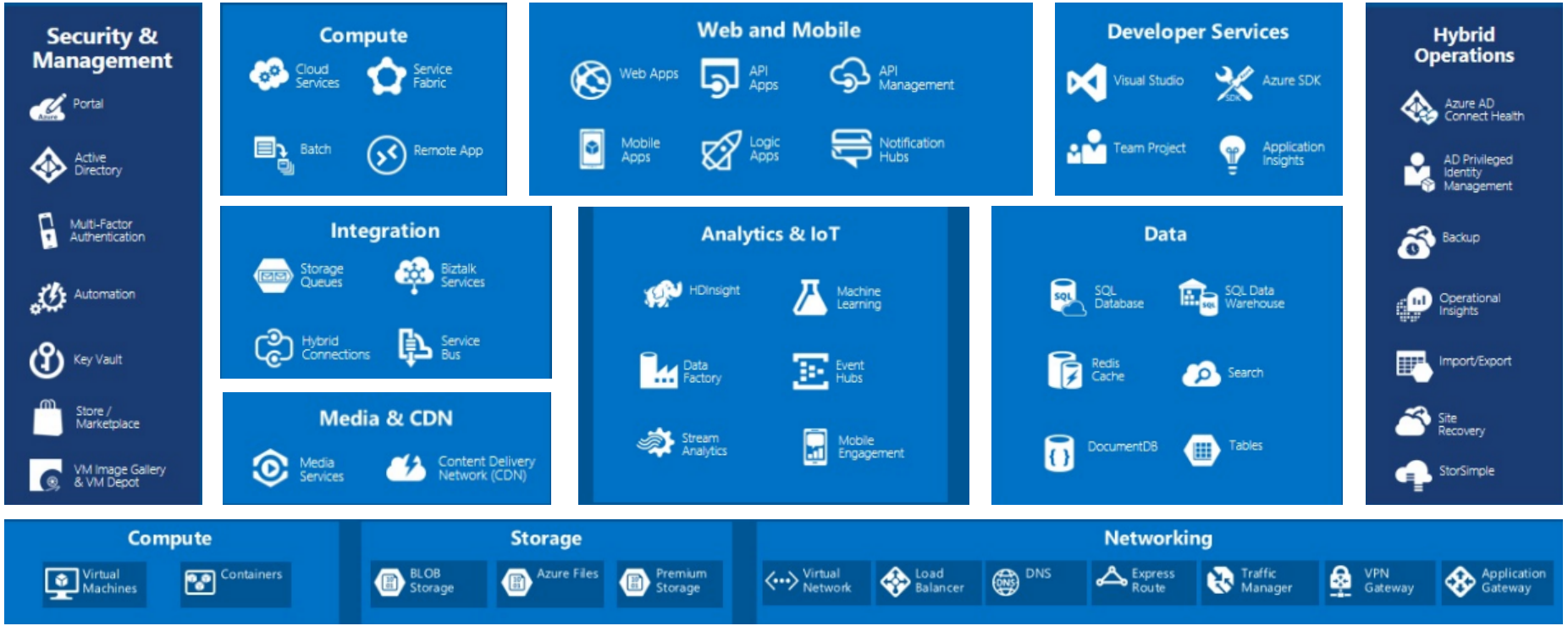
1 SECOND **DELAY IN** PAGE LOAD

- **11%** — **Fewer Page Views**
- **16%** — **Less Customer Satisfaction**
- **7%** — **Loss in Profit**

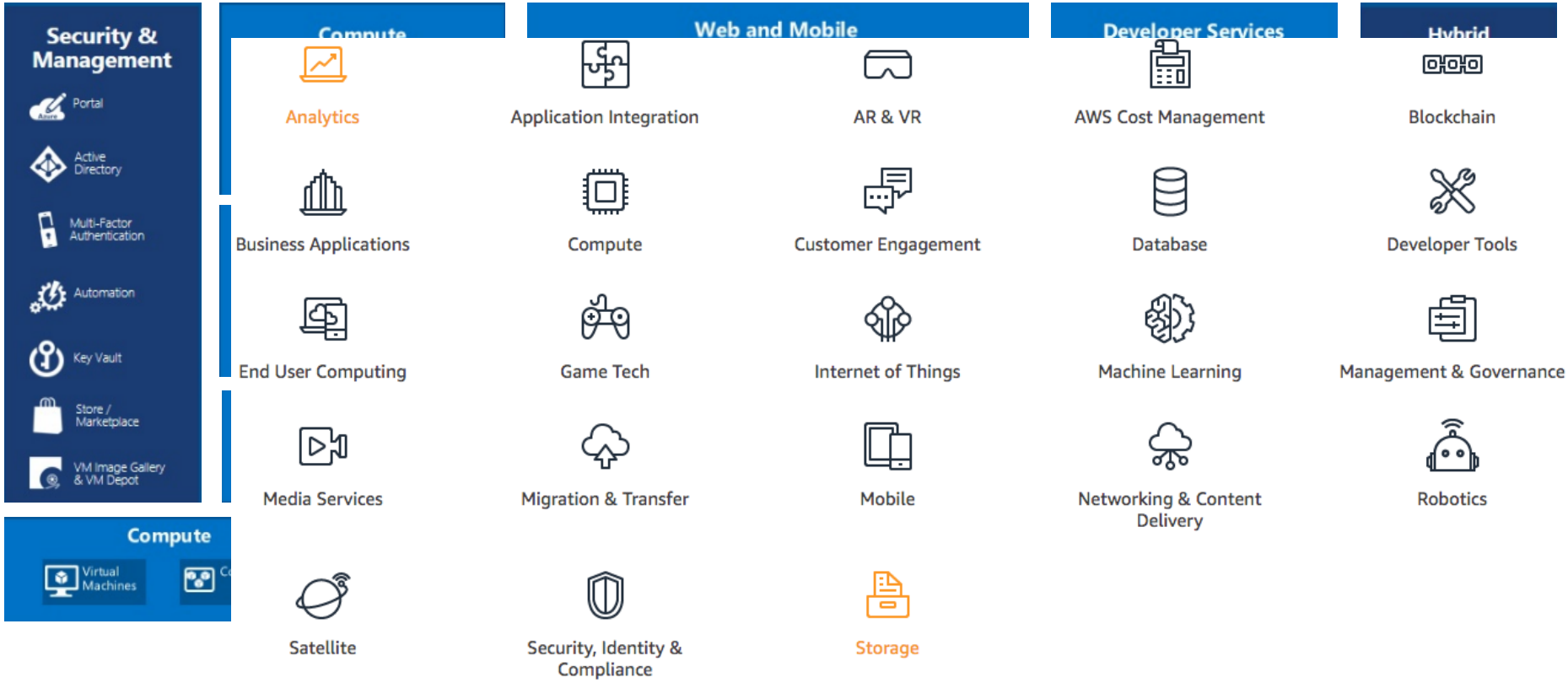
- Low latency is challenging given the data size



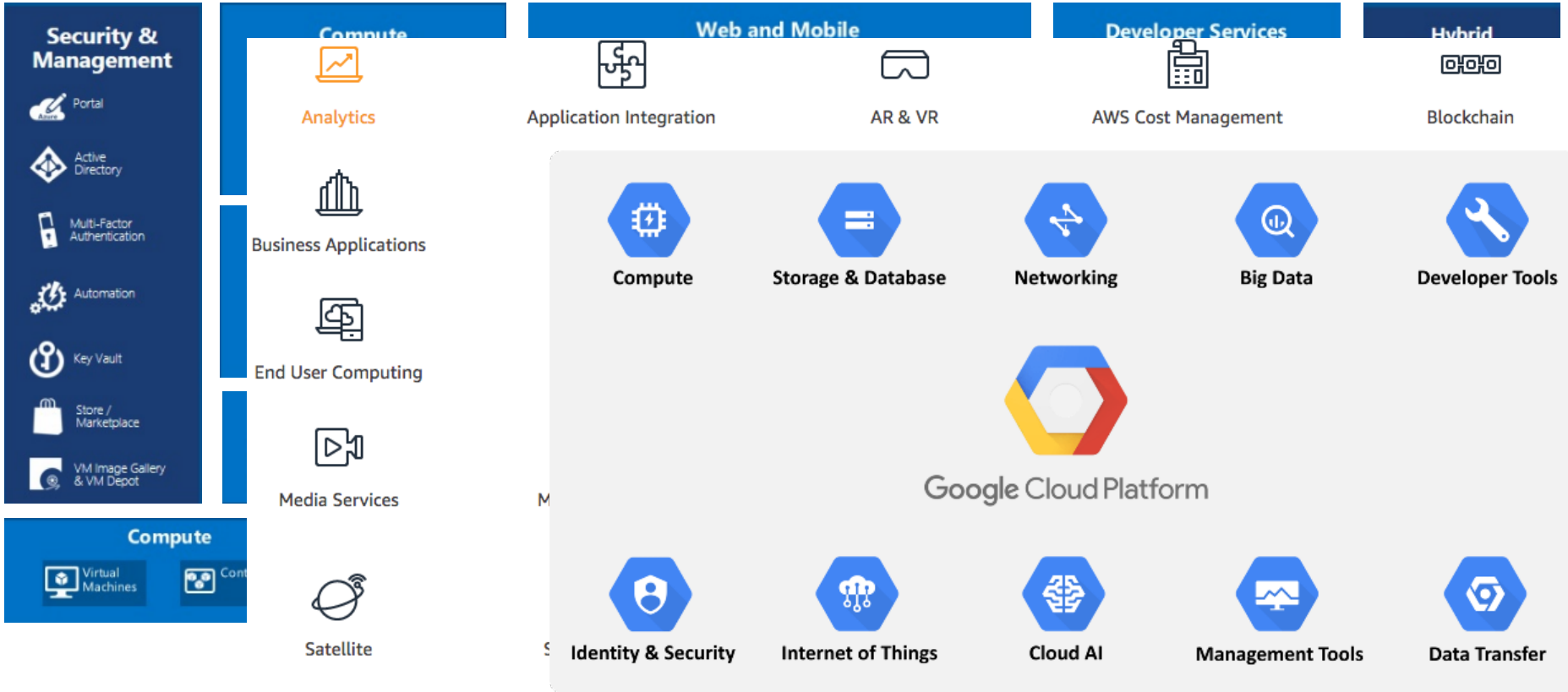
Cloud services



Cloud services in Azure, AWS



Cloud services in Azure, AWS, GCP



Reliability is critical for cloud services

Security & Management
Portal
Active Directory
Multi-Factor Authentication
Automation
Key Vault
Store / Marketplace
VM Image Gallery & VM Depot

Compute
Virtual Machines

Web and Mobile
Application Integration
AR & VR
Customer Engagement
Internet of Things
Mobile
Satellite
Security, Identity & Compliance
Storage

Developer Services
AWS Cost Management
Database
Machine Learning
Networking & Content Delivery

Hybrid
Blockchain
Developer Tools
Management & Governance
Robotics

Relia

Businesses Losing $700 Billion a Year to IT Downtime, Says IHS

Network Interruptions Are the Biggest Culprit

Monday, January 25, 2016 9:01 am EST

Data Centre ▸ Cloud

AWS's S3 outage was so bad Amazon couldn't get into its own dashboard to warn the world

Websites, apps, security cams, IoT gear knackered

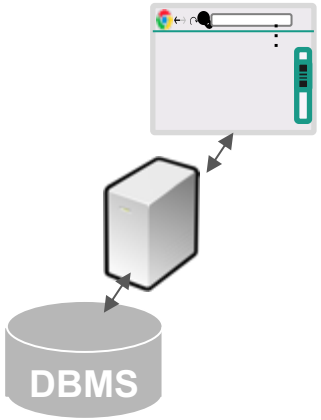
By Shaun Nichols in San Francisco 1 Mar 2017 at 03:00 122 SHARE ▼

Satellite

Security, Identity & Compliance

Storage

Outline



- What slows down (big data) web applications [ICSE'18]
  - What can we do about it? [CIKM'17, FSE'18, ICSE'19, CIDR'20]

    1000+ bugs found

- What stops cloud systems? [HotOS'19]
  - What can we do about it? [ASPLOS'16, ASPLOS'17, ASPLOS'18, PLDI'19, SOSP'19]

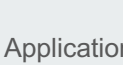
    1000+ bugs found

Microsoft Azure

What Slowed Down Database-Backed Web Applications

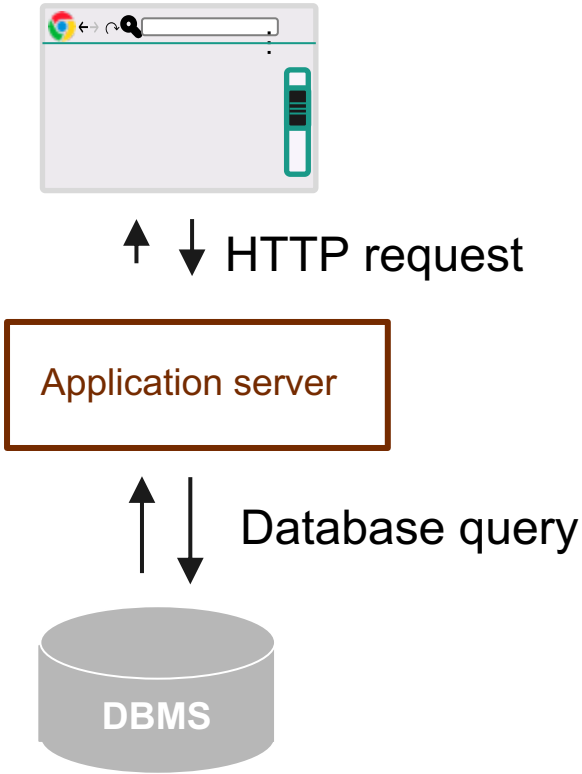
hyperloop.cs.uchicago.edu

Shan Lu

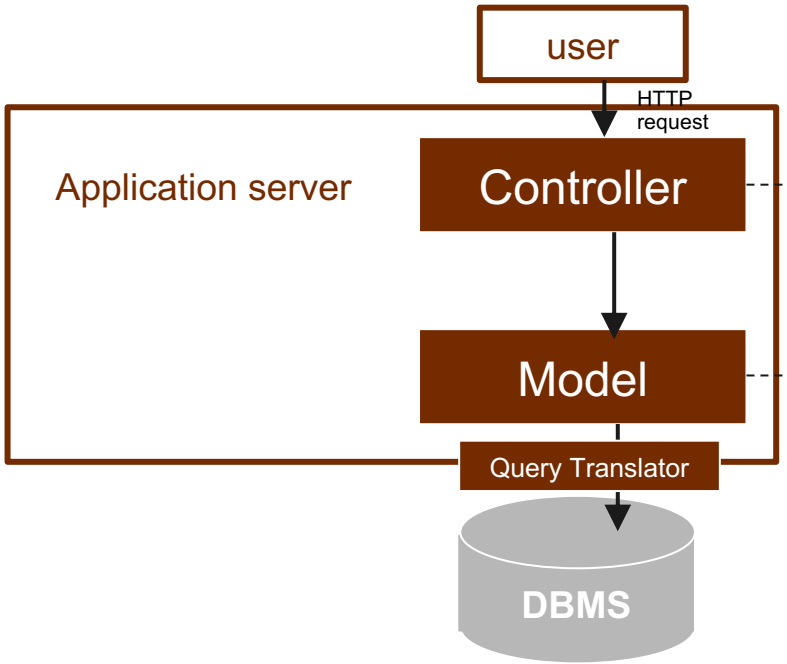


View-Centric Performance Optimization for Database-Backed Web Applications. *ICSE'19*
How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*
PowerStation: Automatically detecting and fixing inefficiencies of database-backed web applications in IDE. *FSE'18*

Common Web-app Architecture



Common Web-app Architecture

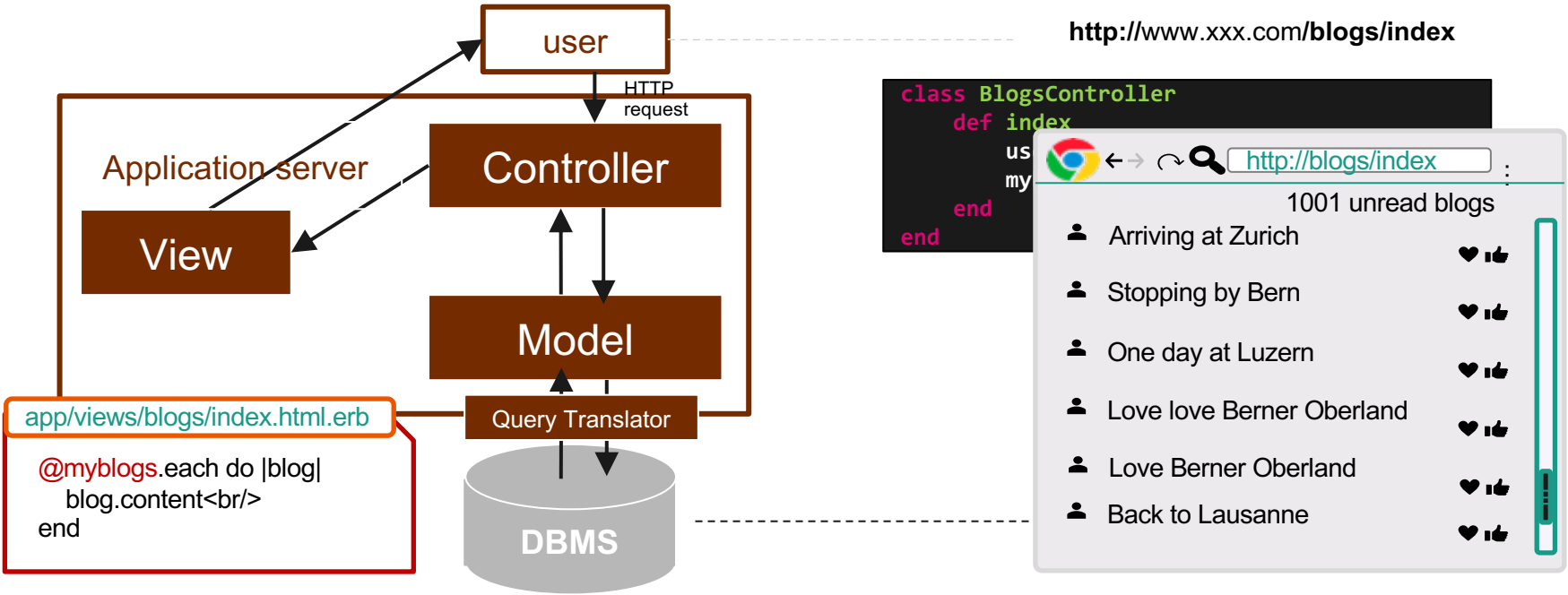


http://www.xxx.com/blogs/index

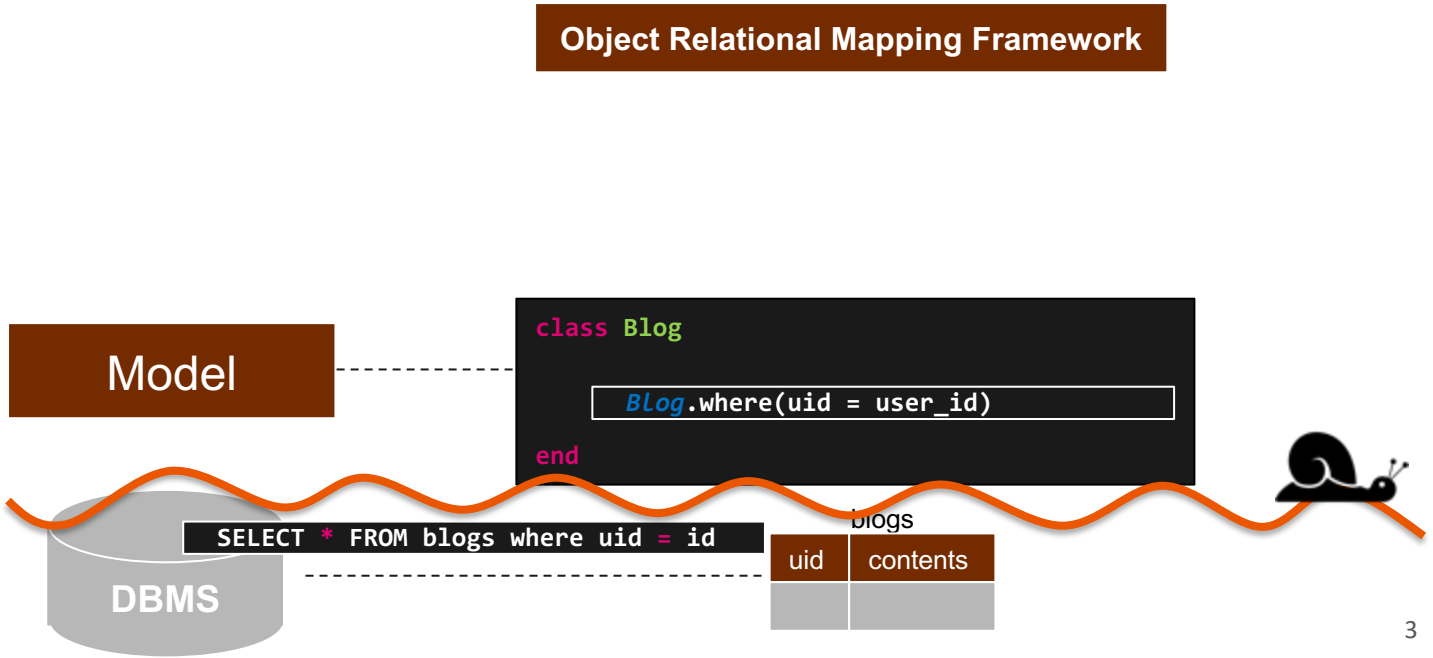
```
class BlogsController
    def index
        user_id = 1
        myblogs = Blog.retrieve(user_id)
    end
end
```

```
class Blog
    def retrieve(user_id)
        SELECT * FROM blogs where uid = id
    end
end
```

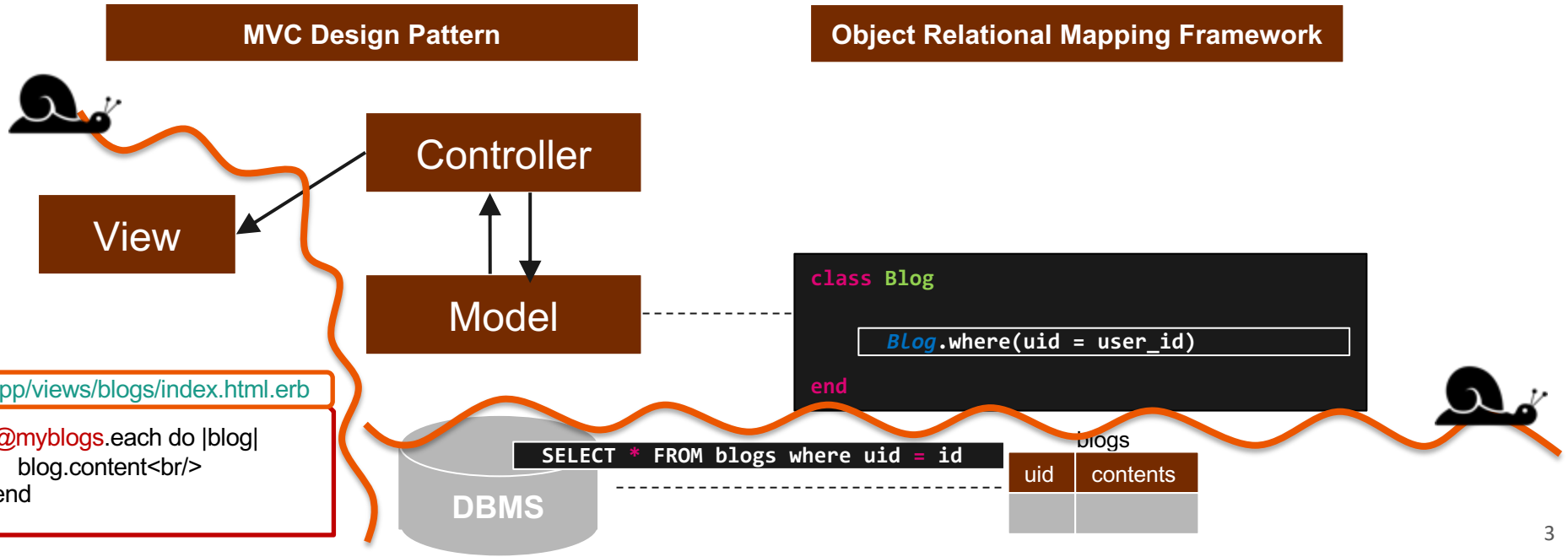
Common Web-app Architecture



Potential sources of inefficiencies



Potential sources of inefficiencies



Outline

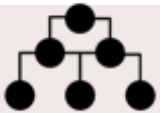
How severe is the problem?

Profile 12 apps from 6 common categories

64 issues in 40 pages

What are the common inefficiency patterns?

Build performance-bug taxonomy



9 anti-patterns

How to solve the problem?

Design automated bug detection & fixing

1000 + bugs

Outline

Profile 12 apps from 6 common categories

64 issues in 40 pages

Build performance-bug taxonomy

Design automated bug detection & fixing

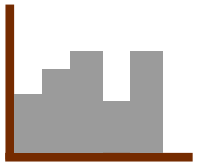
Profiling methodology

Top 2 Apps in 6 popular categories

Table 1: Details of the applications chosen in our study

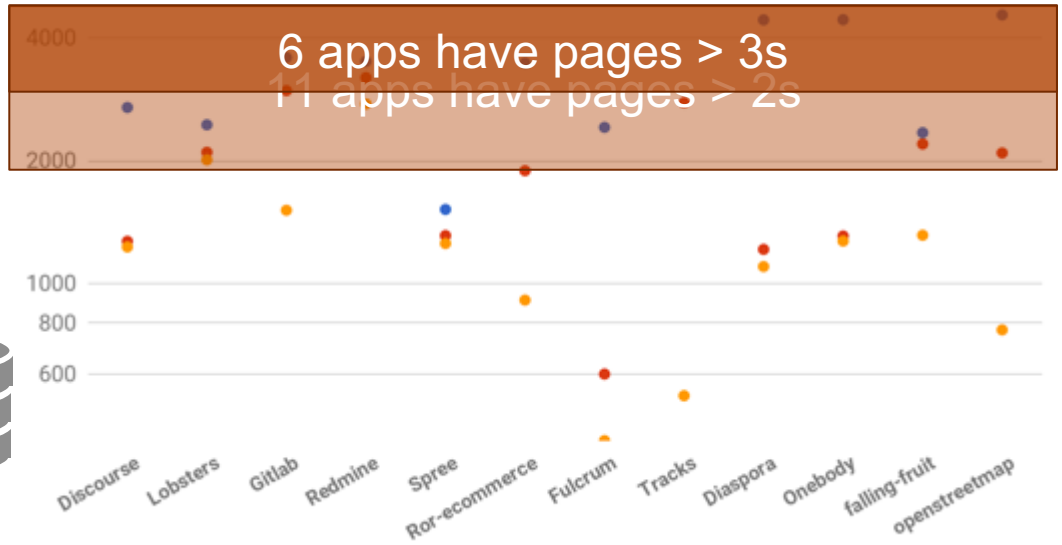
Category	Abbr.	Name	Stars	Commits	Contributors
Forum	Ds	Discourse	21238	22501	568
	Lo	Lobster	1304	1000	48
Collaboration	Gi	Gitlab	19255	49810	1276
	Re	Redmine	2399	13238	6
E-commerce	Sp	Spree	8331	17197	709
	Ro	Ror_ecommerce	1109	1727	21
Task-management	Fu	Fulcrum	1502	697	44
	Tr	Tracks	835	3512	62
Social Network	Da	Diaspora	11183	18734	335
	On	Onebody	1592	1220	6
Map	OS	Openstreetmap	664	8000	112
	FF	Fallingfruit	41	1106	7

Synthesize DB content based on real-world website statistics



How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

Profiling End-to-end Page Time



40 problematic pages
Server takes most time

20000 record

How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

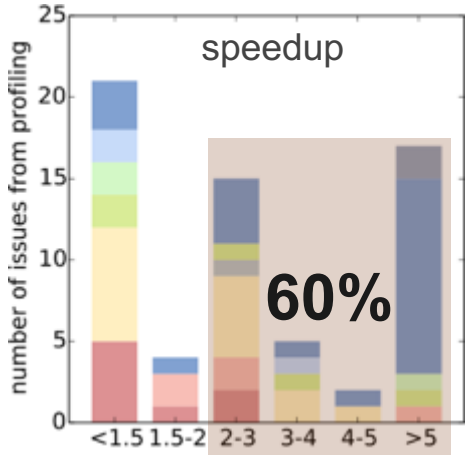
Why is it slow?

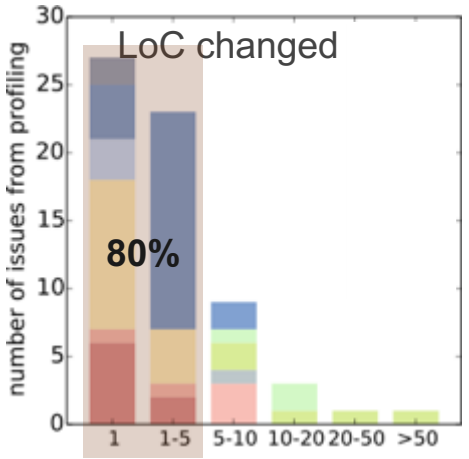
There are inefficiency bugs!

How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

Why is it slow?

- We manually fix the 64 issues we found across 39 pages

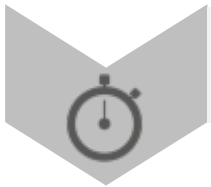




There are bugs!

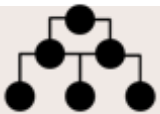
How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

Outline



Profile 12 apps from 6 common categories

Build performance-bug taxonomy



9 anti-patterns

Design automated bug detection & fixing

How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

Common Performance Anti-patterns

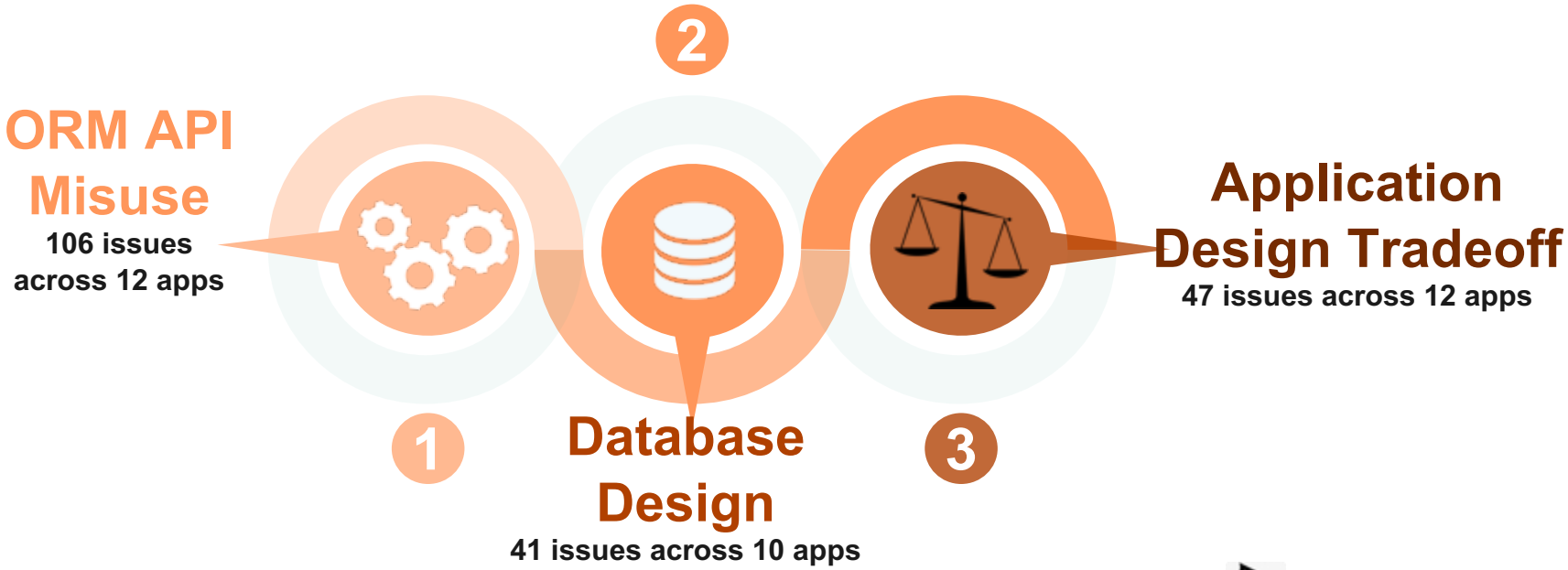
64 performance issues from profiling

140 performance issues from bug tracking system

9 anti-patterns

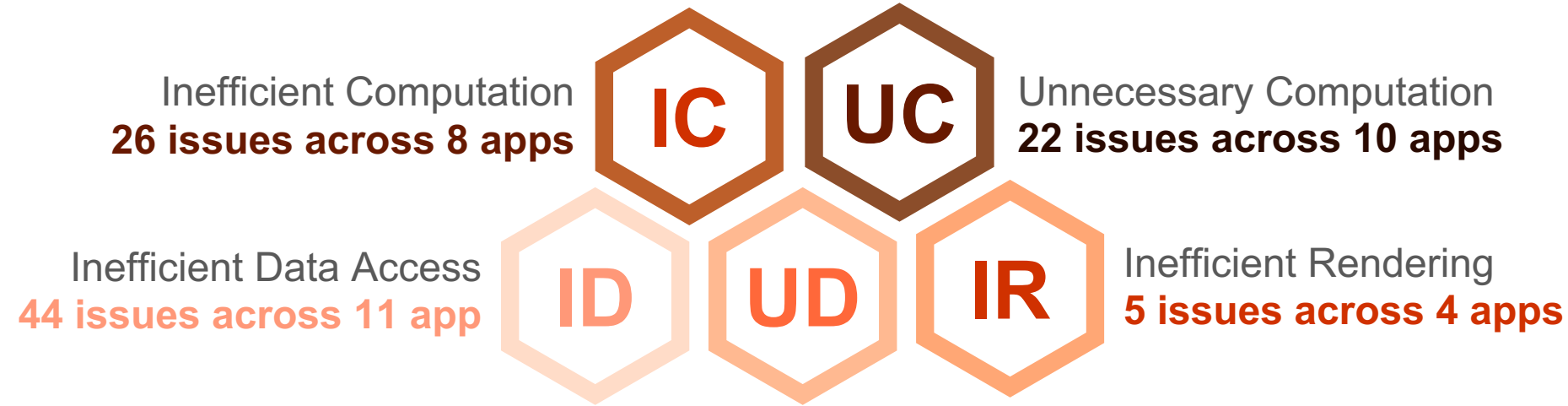
How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

Common Performance Anti-patterns



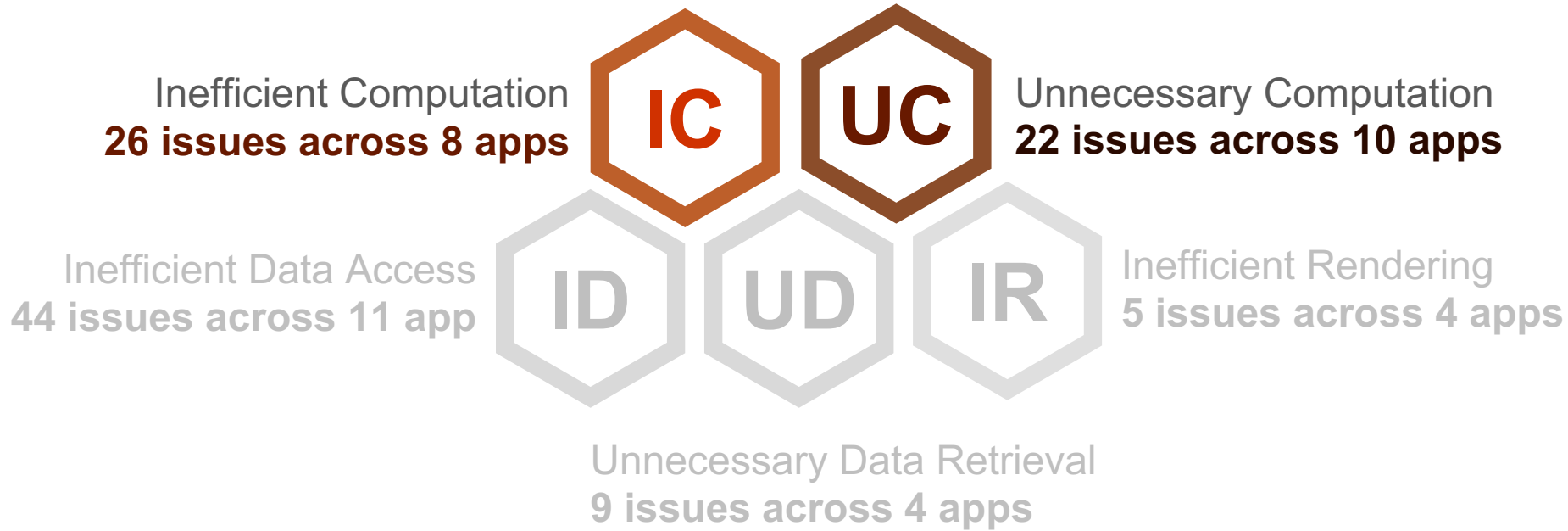
How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

ORM API Misuse



How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

ORM API Misuse



How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

ORM API Misuse: inefficient computation

project.issues.count>0	**SELECT** COUNT(*) **FROM** issues **WHERE** project_id = ?	inefficient
project.issues.any?	**SELECT** COUNT(*) **FROM** issues **WHERE** project_id = ?	inefficient
project.issues.exists?	**SELECT** 1 AS ONE **FROM** issues **WHERE** project_id = ? LIMIT 1	efficient

spree

2X speedup

How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

ORM API Misuse: unnecessary computation


```
values.each do |value|
  u.issues.include? value
end
```

How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

ORM API Misuse: unnecessary computation

```
  values.each do |value|
-   u.issues.include?value
  end
```

```
+   rans = u.issues
  values.each do |value|
+   rans.include?value
  end
```

REDMINE

20X speed up

How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

ORM API misuses that affect memory consumption

- *map (:id)* VS *pluck (:id)*
- pluck(size).sum VS sum(size)
- *pluck + pluck* VS *SQL UNION*
- *…*

How to tackle API Misuses?

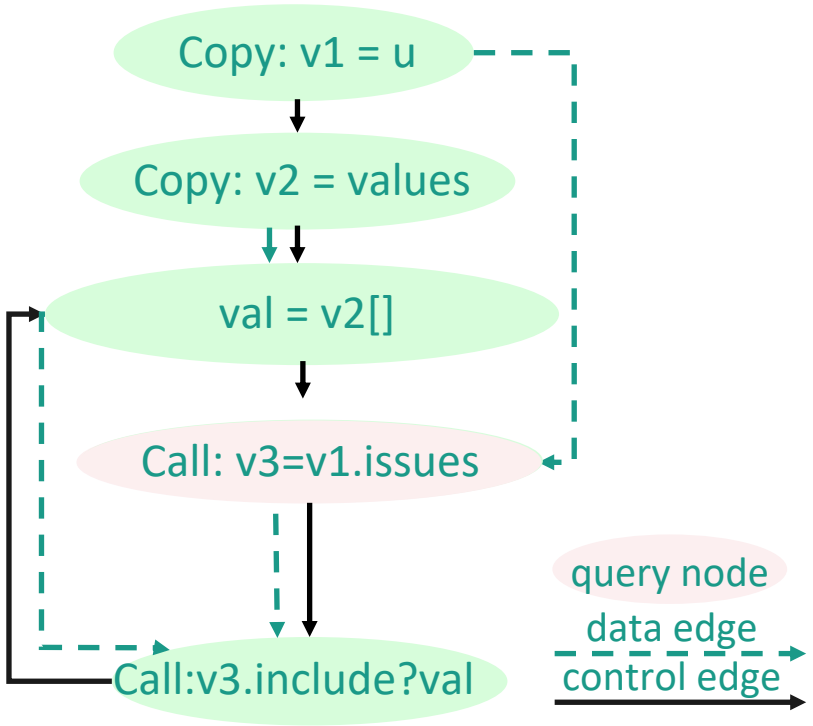
- Why cannot existing compiler handle this?

- Can we extend compiler to
  - Understand ORM APIs and queries?
  - Detect the problem?
  - Solve the problem?

Database-aware PDG

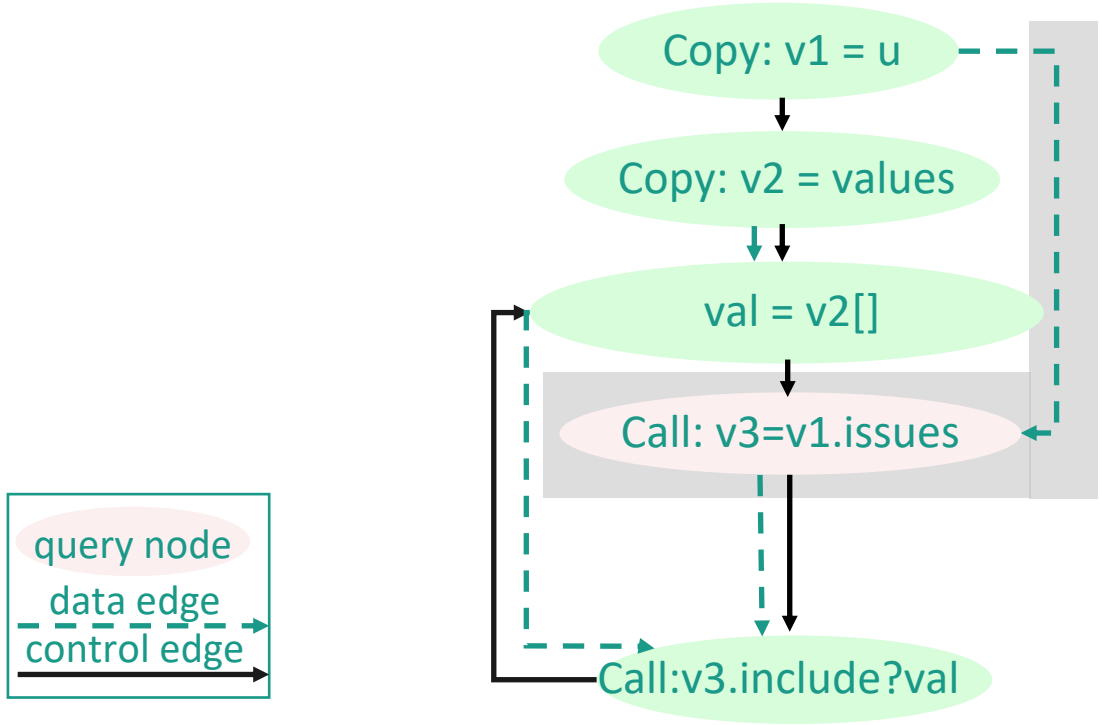
```
v1 = u
v2 = values
values.reject |val|
  u.issues.include?val
end
end
```

```
SQL: SELECT * from issues
                WHERE user_id=?
```



(b) PDG

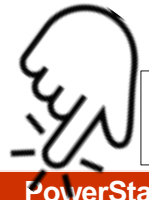
Detect and Fix



Loop-invariant query

PowerStation: Automatically detecting and fixing inefficiencies of database-backed web applications in IDE. *FSE'18*

PowerStation (Integrated with RubyMine)



RubyMine File Edit View Navigate Code Refactor Run Tools VCS Window Help

PowerStation
Whole App
Single Action

blog [~/Research/blog] - .../app/controllers/blogs_controller.rb [blog]

blog app controllers blogs_controller.rb

blogs_controller.rb

```
1  class IssuesController < ApplicationController
2    def read_only_fields
3      ran = run_query
4      values.reject do |val|
5        ran.include?val
6      end
7    end
8  end
```

PowerStation | issues

LI | IA | CS | IR | DS | RD

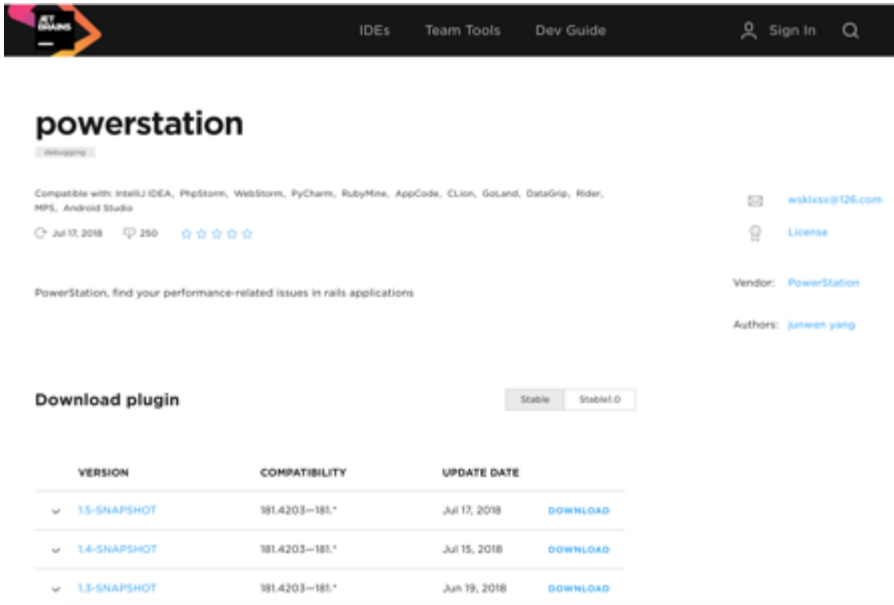
blogs_controller.rb 4 | FIX

Issue List

PowerStation: Automatically detecting and fixing inefficiencies of database-backed web applications in IDE.

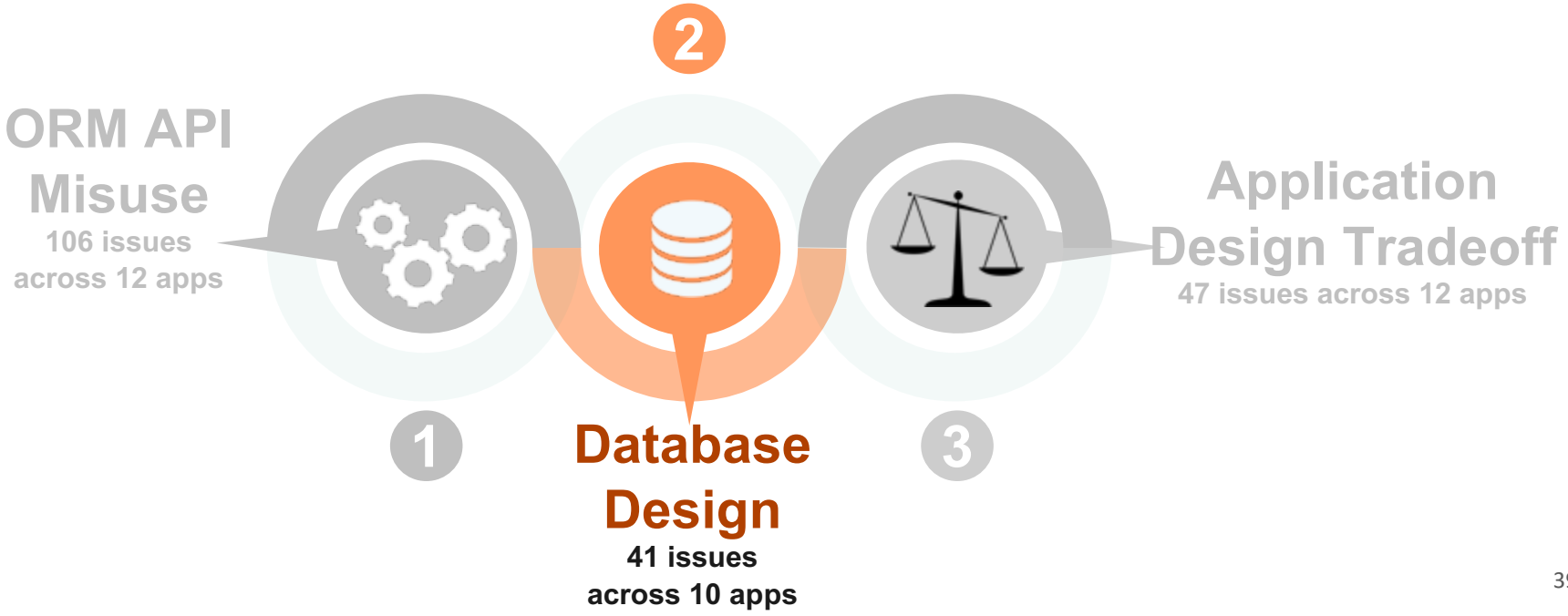
Try our Powerstation!

- 12 real world apps
- 1221 inefficiencies found



PowerStation: Automatically detecting and fixing inefficiencies of database-backed web applications in IDE. *FSE'18*

Common Performance Anti-patterns



Database Design Problem

- Missing fields (8 issues across 5 apps):
  - fields derivable from other fields and not persistently stored

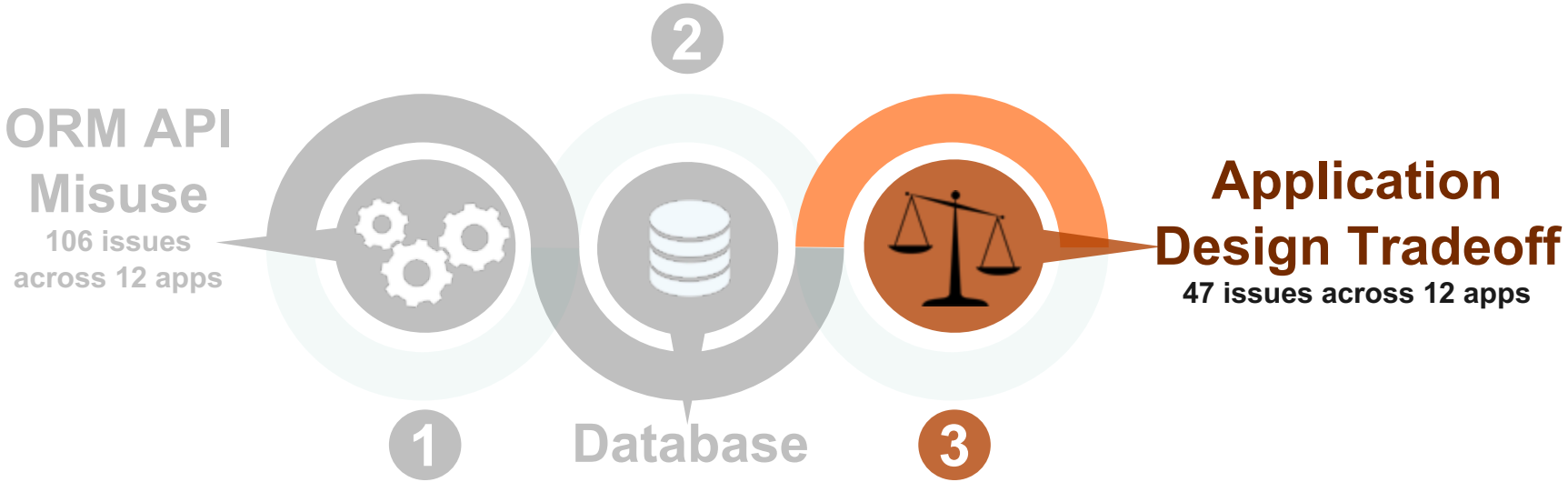
id	longitude	latitude	location

2X

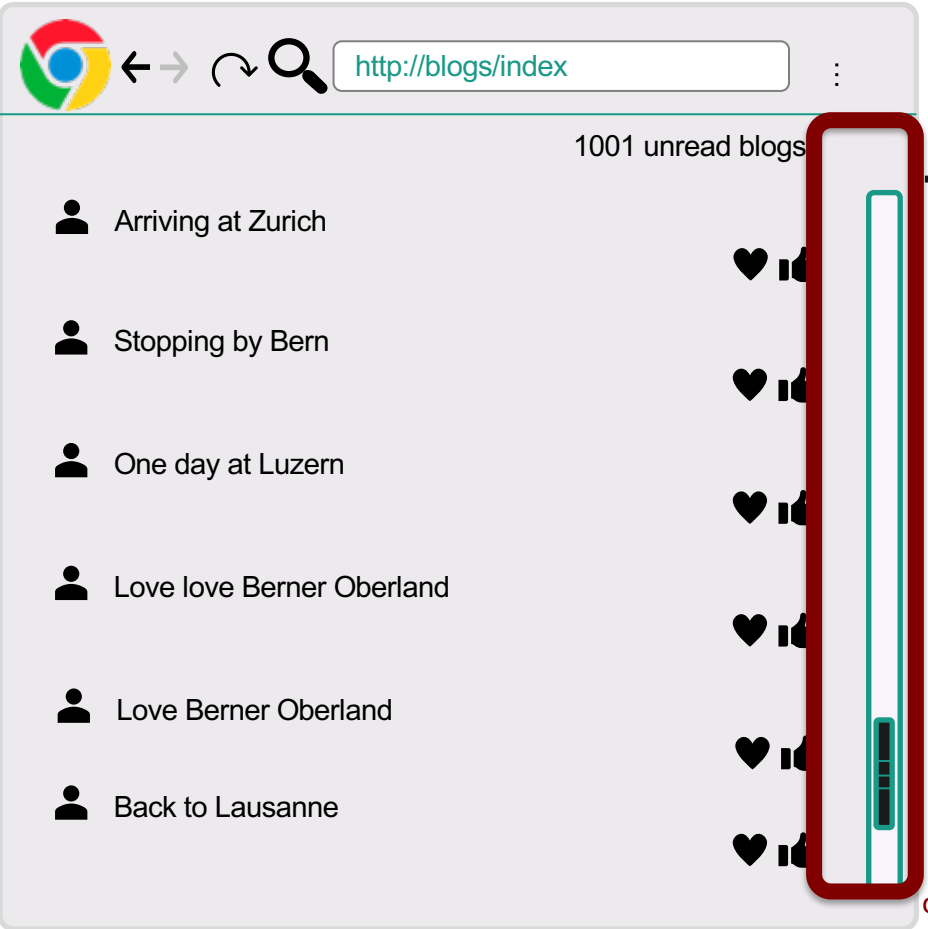
- Missing index (33 issues across 10 apps)

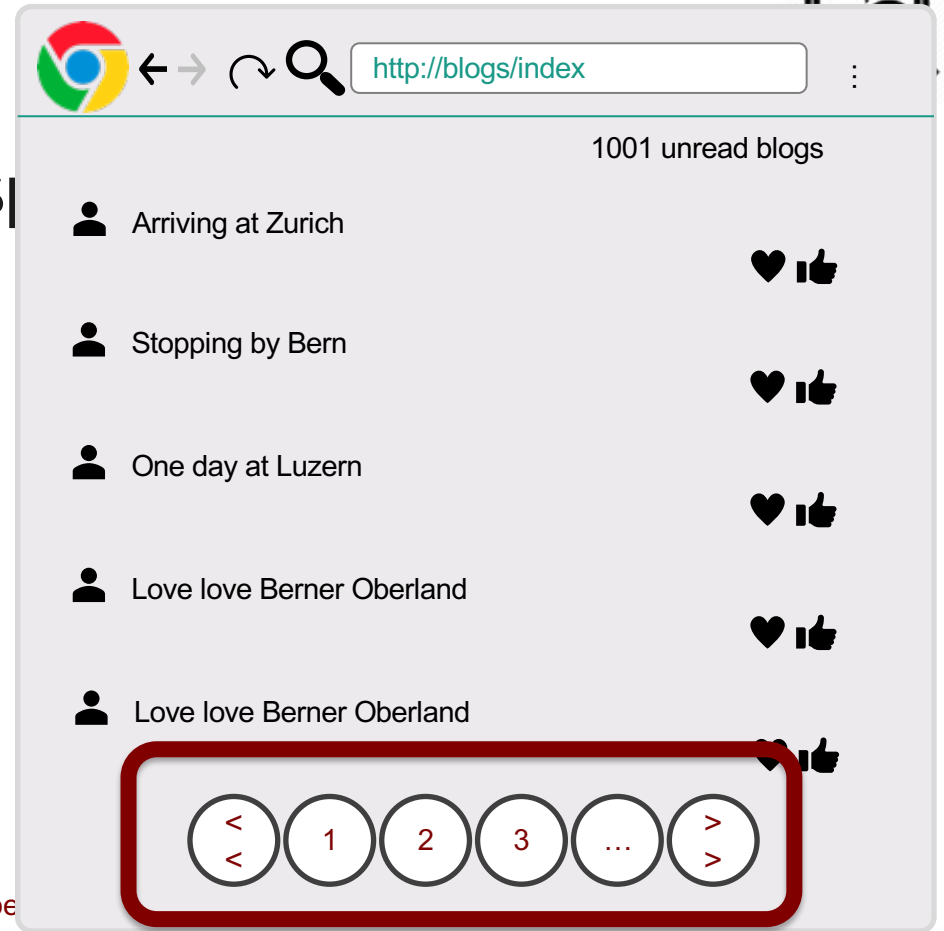
How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*

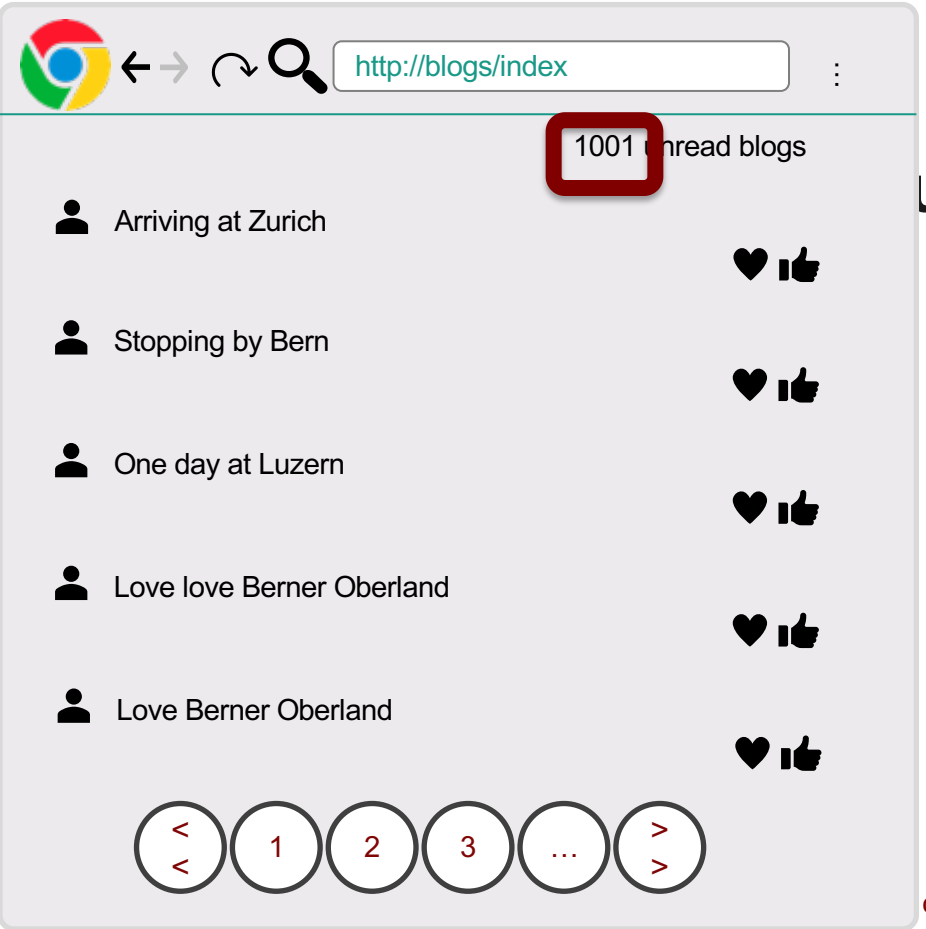
Common Performance Anti-patterns

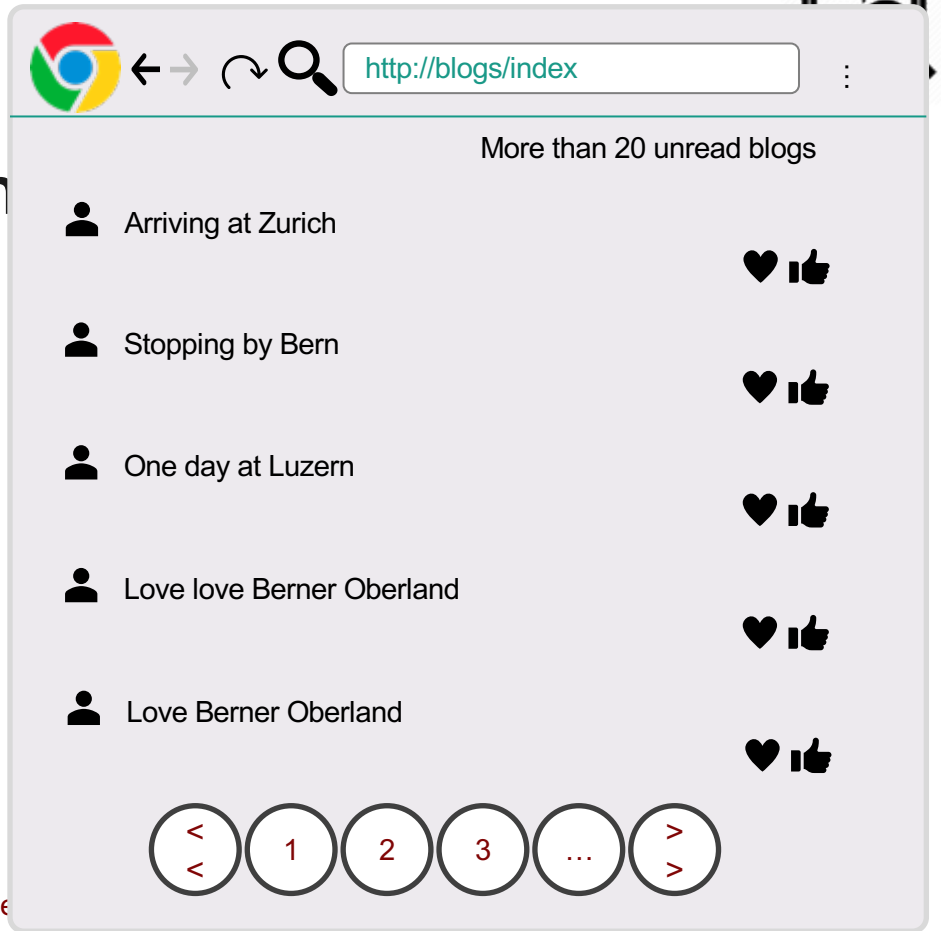


How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*


http://blogs/index
1001 unread blogs
Arriving at Zurich
Stopping by Bern
One day at Luzern
Love love Berner Oberland
Love Berner Oberland
Back to Lausanne


http://blogs/index
1001 unread blogs
Arriving at Zurich
Stopping by Bern
One day at Luzern
Love love Berner Oberland
Love love Berner Oberland
< <
1
2
3
...
> >


http://blogs/index
1001 unread blogs
Arriving at Zurich
Stopping by Bern
One day at Luzern
Love love Berner Oberland
Love Berner Oberland
< <
1
2
3
...
> >


http://blogs/index
More than 20 unread blogs
Arriving at Zurich
Stopping by Bern
One day at Luzern
Love love Berner Oberland
Love Berner Oberland
< <
1
2
3
...
> >


Home Recent Comments Your Threads Submit Story Search
Filters 20+ New Messages test (0) Logout
Submit a Story
URL:
Fetch Title
Title:
Tags:
Text:
Optional when submitting a URL; please see guidelines
SELECT count(*) FROM moderations JOIN stories where stories.user_id = @user.id AND moderations.created_at > 5.days.ago
Story submission guidelines
To be able to easily submit a page you're viewing in your browser to Example News, drag the bookmarklet to the right to your bookmark bar. You'll be taken to this page with the viewed page's URL and title.
Submit to Example News
When submitting a URL, the text field is optional and should only be used when additional context or explanation of the URL is needed. Commentary or opinion should be reserved for a comment, so
Whether to show this guideline
Do not editorialize story titles, but when the original story's title has no context or is unclear, please change it. Please remove extraneous components from titles such as the name of the site or section.
If no tags clearly apply to the story you are submitting, chances are it does not belong here. Do not overreach with tags if they are not the primary focus of the story.
When the story being submitted is more than a year or so old, please add the year the story was written to the post title in parentheses.
Author:
I am the author of the story at this URL (or this text)
Submit
Preview
Markdown formatting available
>1.5s

How to tackle application design tradeoffs?

- Can we do automated optimization?

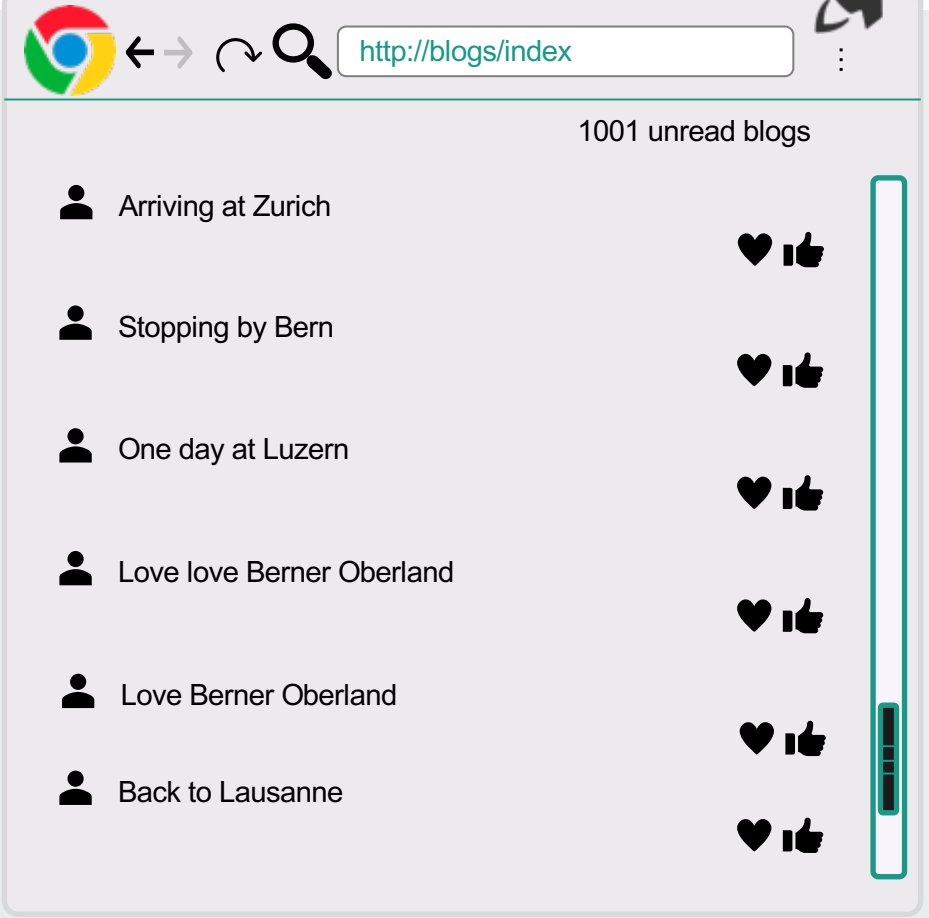
- Help developers make informed decision, by providing
  - Cost information
  - Alternative display/functionality options

app/controllers/blogs_controller.rb

```
def index
    @blogs = blog.all
    render “index”
end
```

app/views/blogs/index.html.erb

```
@blogs.each do |blog|
    blog.content<br/>
end
```

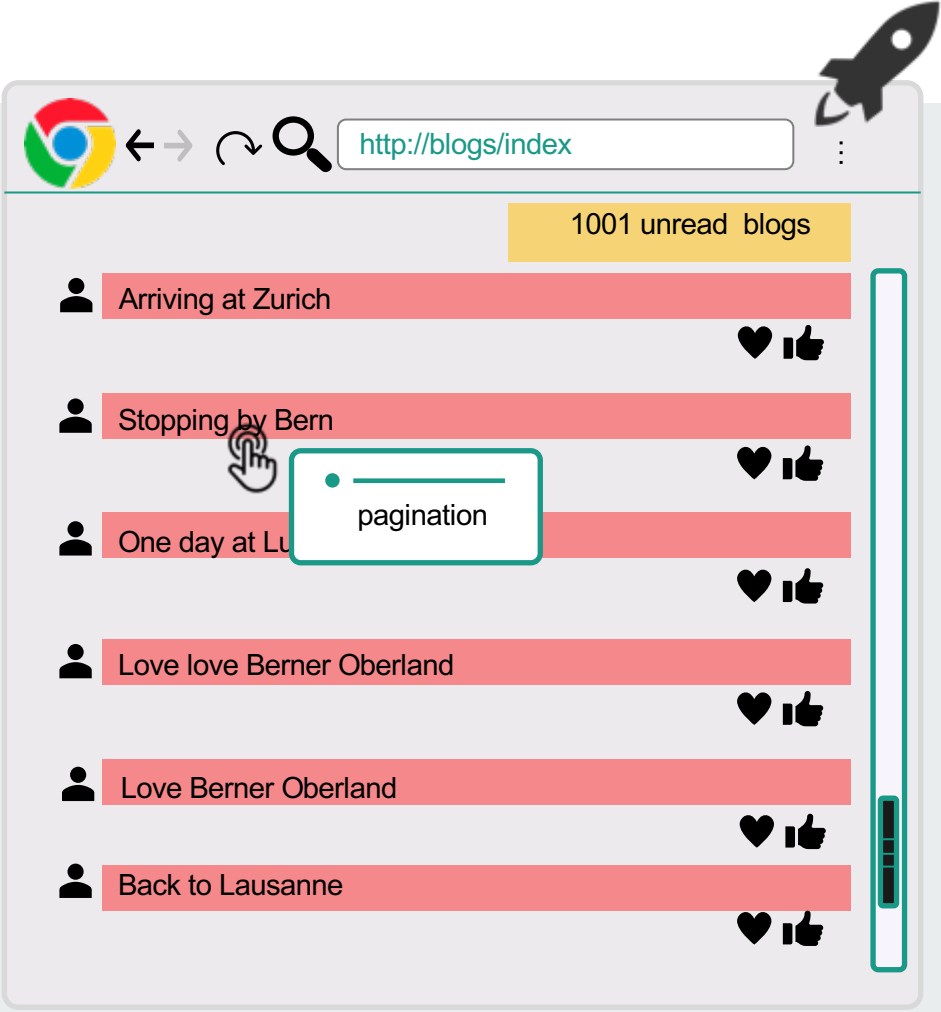


app/controllers/blogs_controller.rb

```
def index
    @blogs = blog.all
    render “index”
end
```

app/views/blogs/index.html.erb

```
@blogs.each do |blog|
    blog.content<br/>
end
```

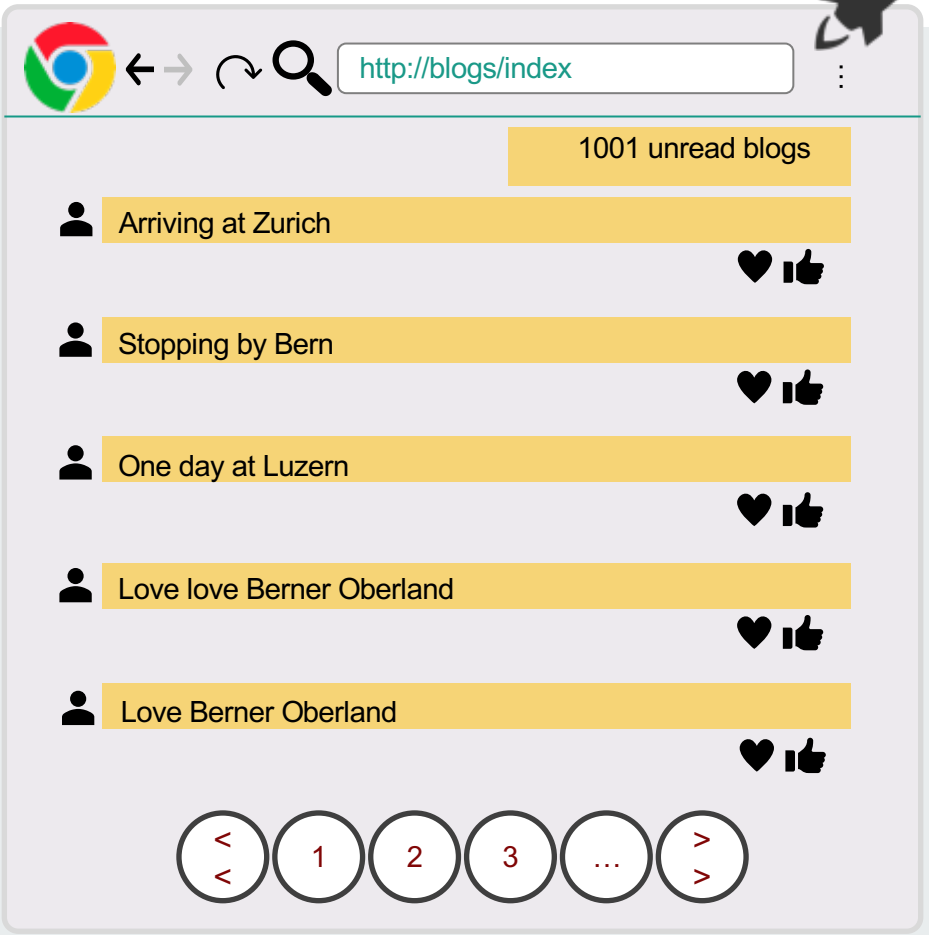


app/controllers/blogs_controller.rb

```
def index
  @blogs = Blog.all.paginate(…)
  render “index”
end
```

app/views/blogs/index.html.erb

```
@blogs.each do |blog|
  blog.content<br/>
end
will_paginate @blogs
```

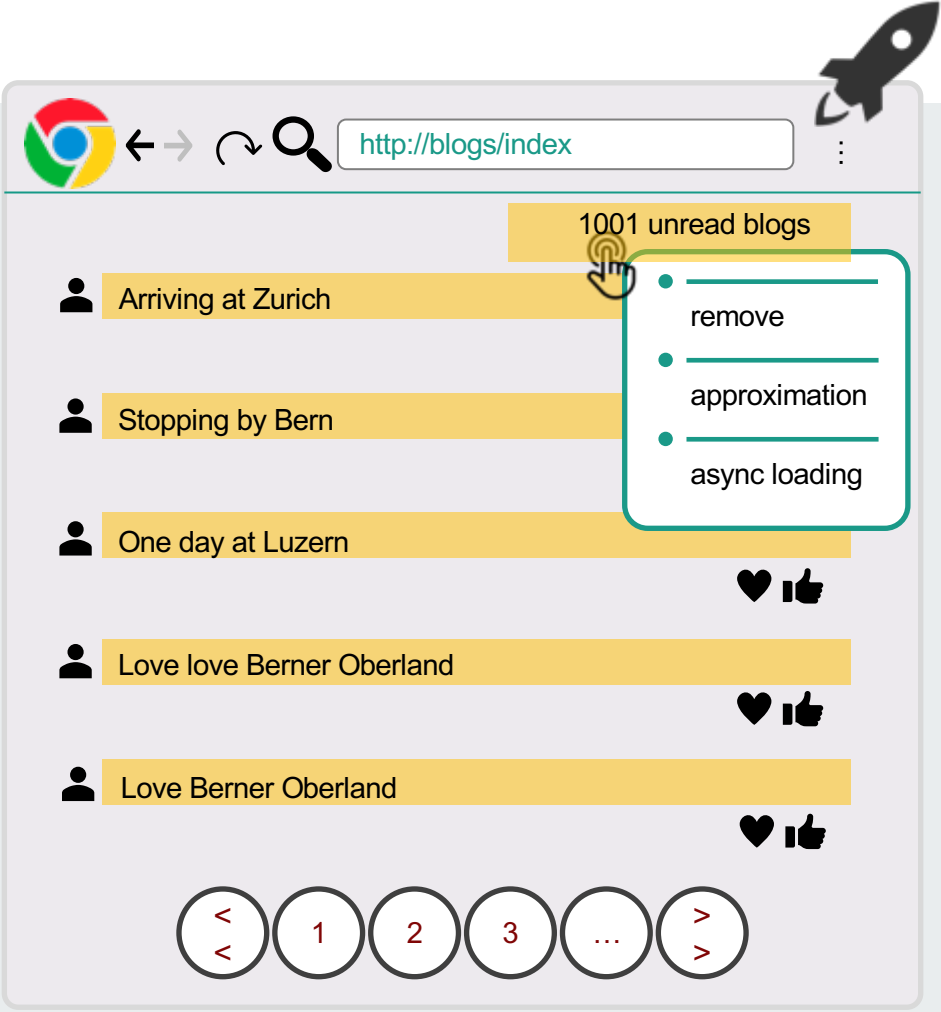


app/controllers/blogs_controller.rb

```
def index
    @blognum = blog.count
    render “index”
end
```

app/views/blogs/index.html.erb

There are @blognum blogs

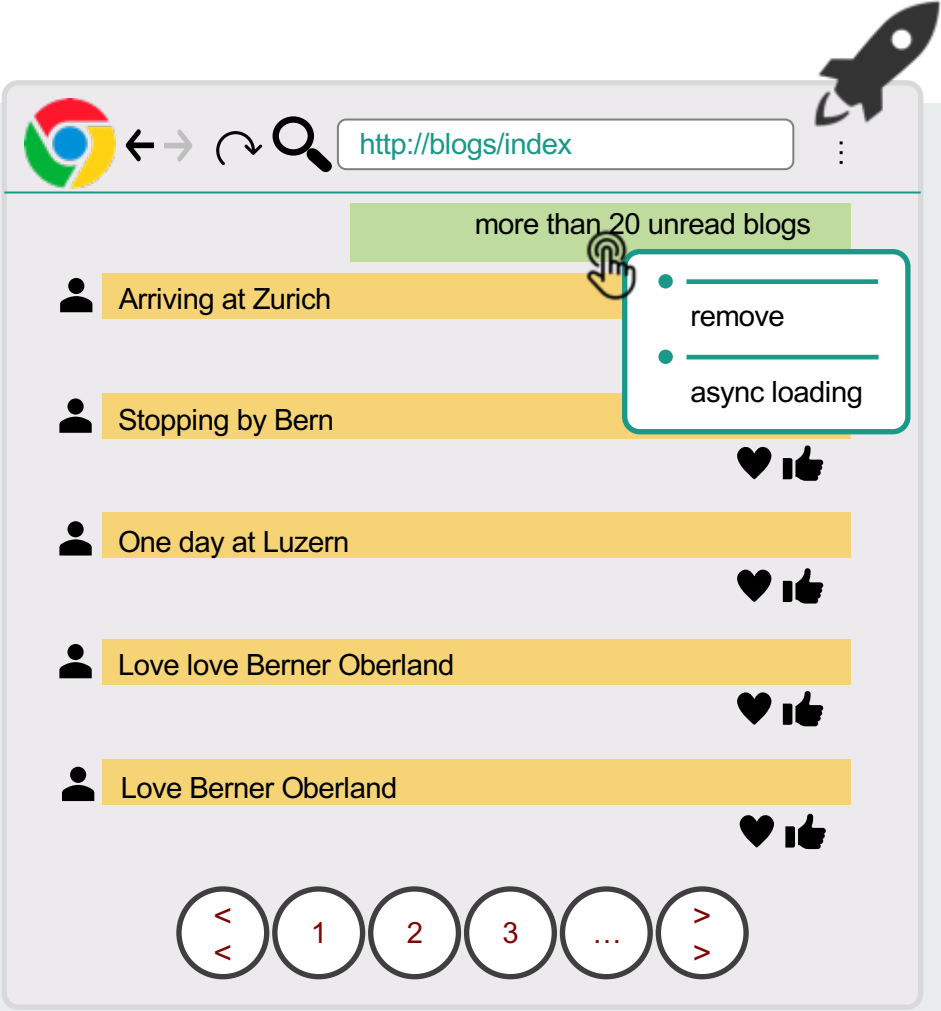


app/controllers/blogs_controller.rb

```
def index
    @blognum = blog.limit(21).count
    render “index”
end
```

app/views/blogs/index.html.erb

```
There are
@blognum>20?‘more than 20’:@blognum
blogs
```

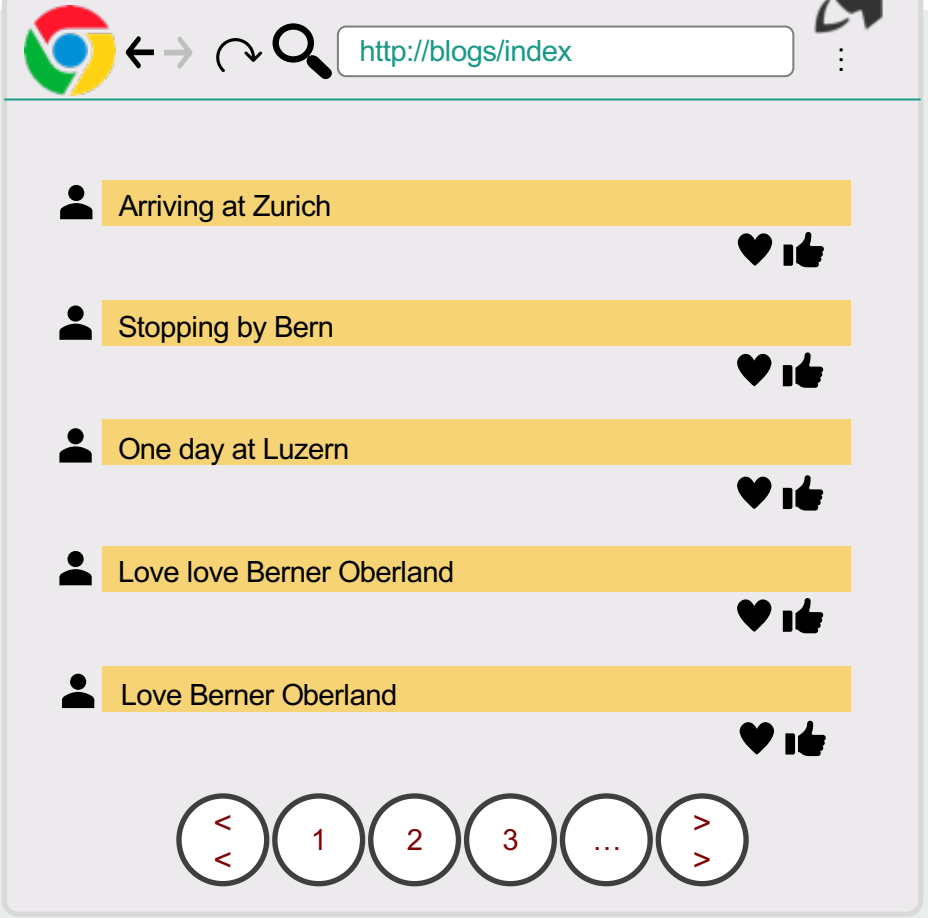


app/controllers/blogs_controller.rb

```
def index
  @blognum = blog.count
  render “index”
end
```

app/views/blogs/index.html.erb

```
@blognum unread blogs
```

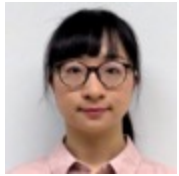


Try our Panorama!

- 12 real world apps
- 149 performance-enhancing opportunities identified for 119 costly HTML tags
- 4.5X average page-load time speedup
- User study agrees!

hyperloop.cs.uchicago.edu

Slow downs in web applications



Junwen Yang

Real world database-backed applications perform poorly

Data-related performance anti-patterns exist

Automatic tools are built to detect and fix performance issues

View-Centric Performance Optimization for Database-Backed Web Applications. *ICSE'19*
How not to structure your database-backed web applications: a study of performance bugs in the wild. *ICSE'18.*
PowerStation: Automatically detecting and fixing inefficiencies of database-backed web applications in IDE. *FSE'18*

What stopped cloud services?

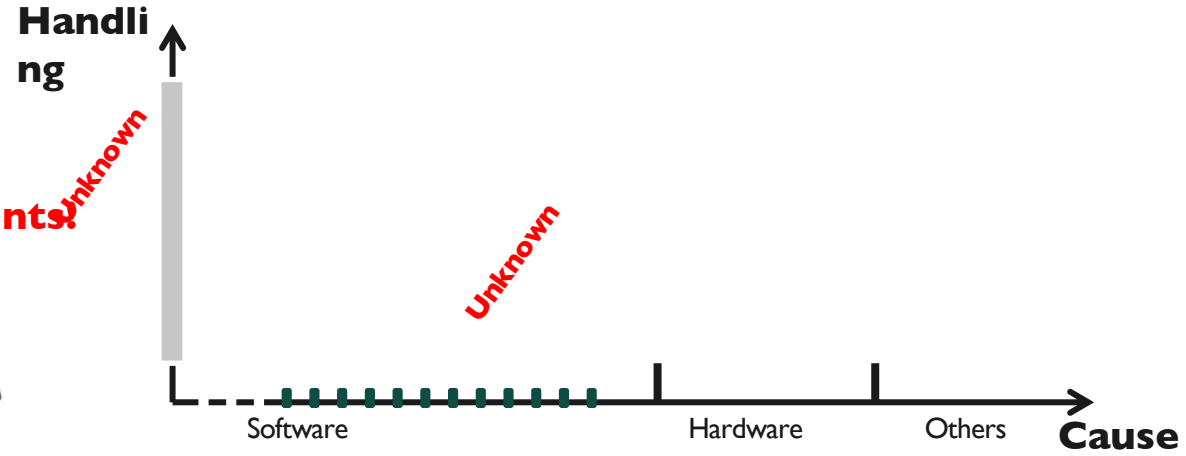
Efficient and Scalable Thread-Safety Violation Detection --- Finding thousands of concurrency bugs during testing. *SOSP'19*
DFix: Automatically Fixing Timing Bugs in Distributed Systems. *PLDI'19*
FCatch: Automatically detecting time-of-fault bugs in cloud systems. *ASPLOS'18*
DCatch: Automatically Detecting Distributed Concurrency Bugs in Cloud Systems. *ASPLOS'17*
TaxDC: A Comprehensive Taxonomy of Non-Deterministic Concurrency Bugs in Cloud Distributed Systems. *ASPLOS'16.*
What Bugs Cause Production Cloud Incidents? *HotOS'19*

UCHICAGO

Microsoft

Need to study real-world cloud incidents

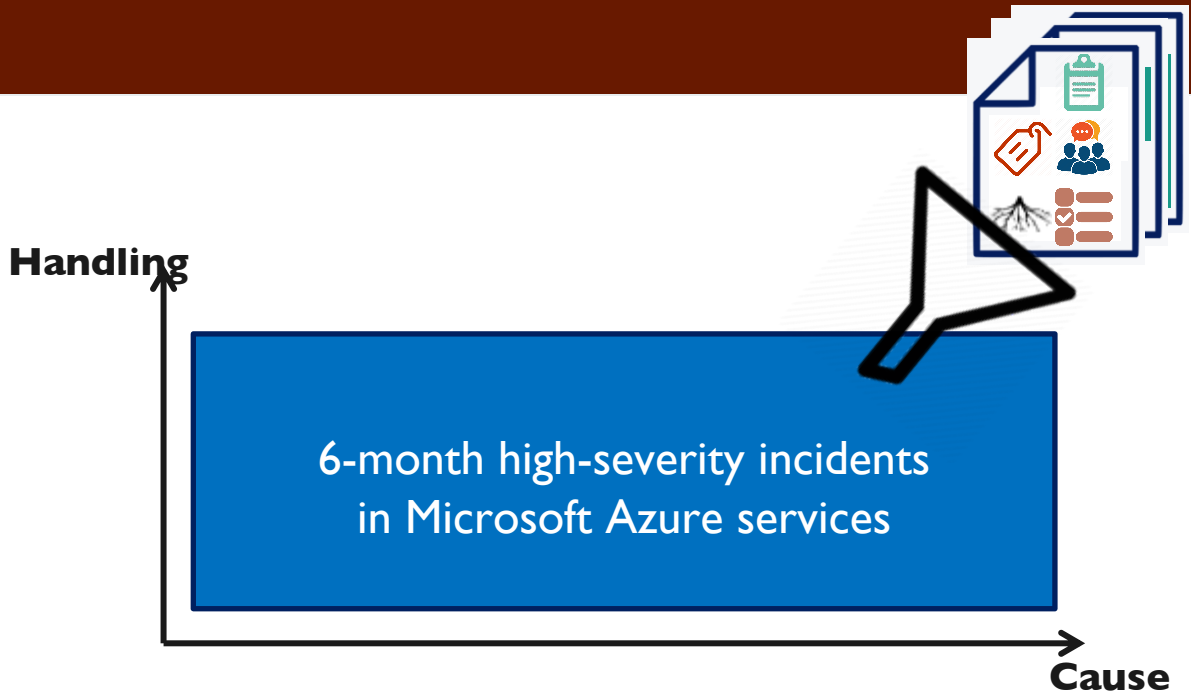
Existing studies for cloud incidents



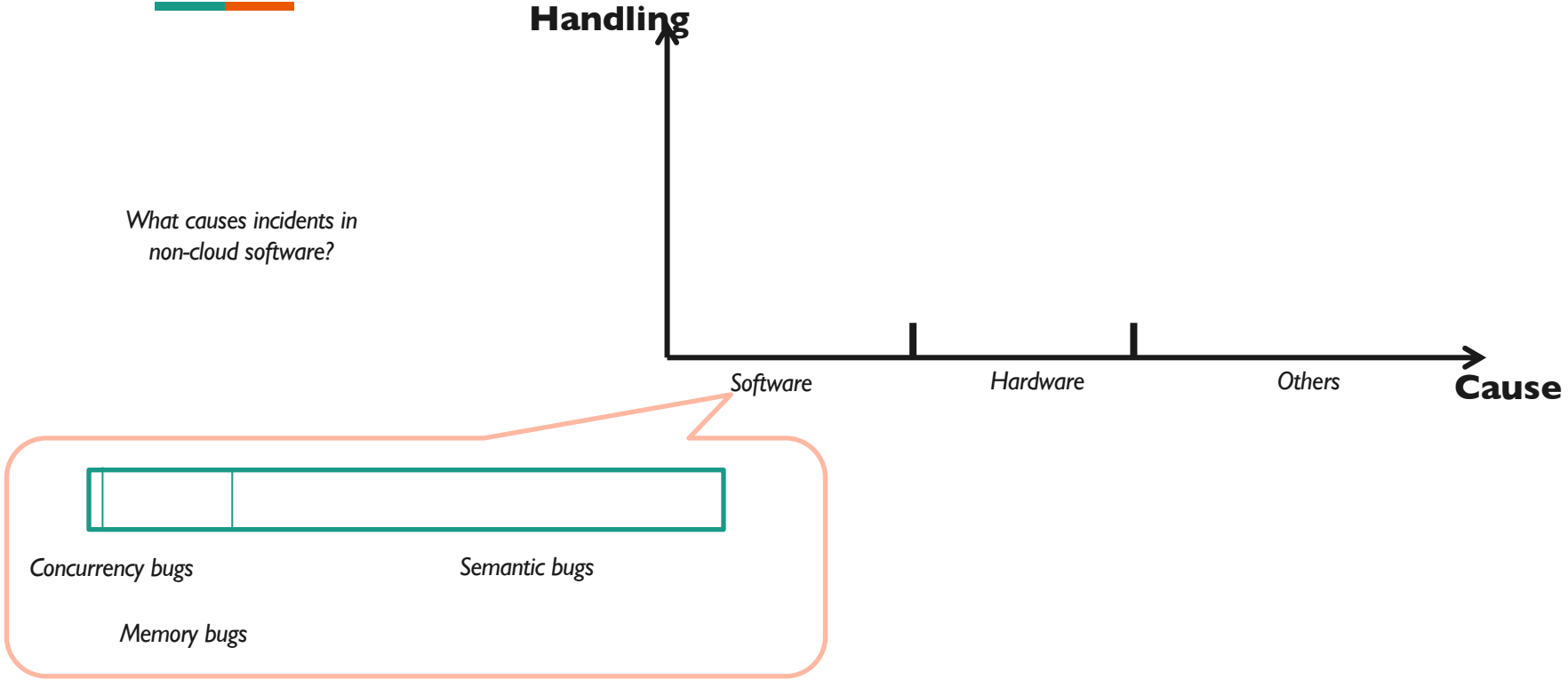
Data source constraints!

[6] Leesatapornwongsa. TaxDC. In ASPLOS'16
[5] Leesatapornwongsa. Scalability bugs. In HotOS'17
[4] Huang. Gray failure. In HotOS'17
[3] Yuan. Simple test can prevent most critical failures. In OSDI'14
[2] Gunawi. Why does the cloud stop computing? In SoCC'16
[1] Gunawi. What bugs live in the cloud? In SoCC'14

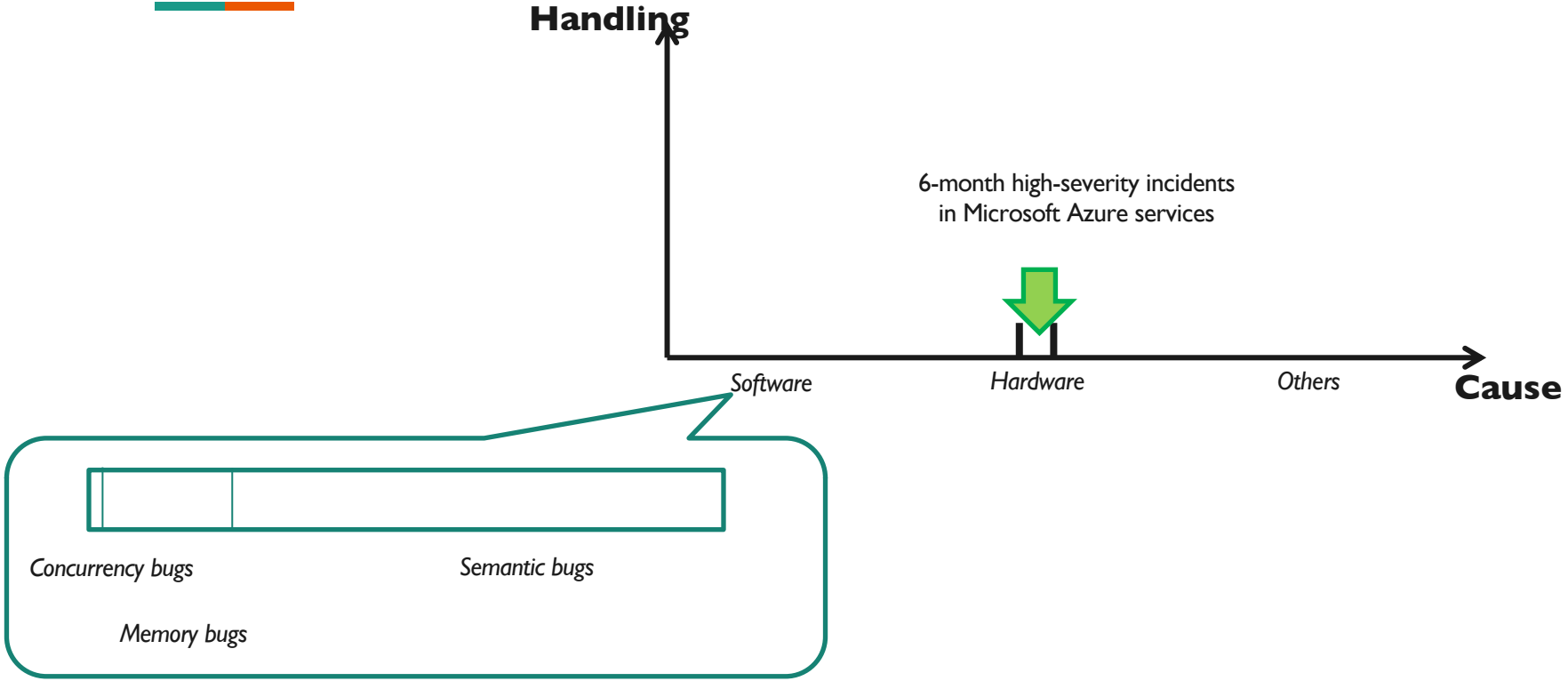
Our work


Handling
6-month high-severity incidents
in Microsoft Azure services
Cause

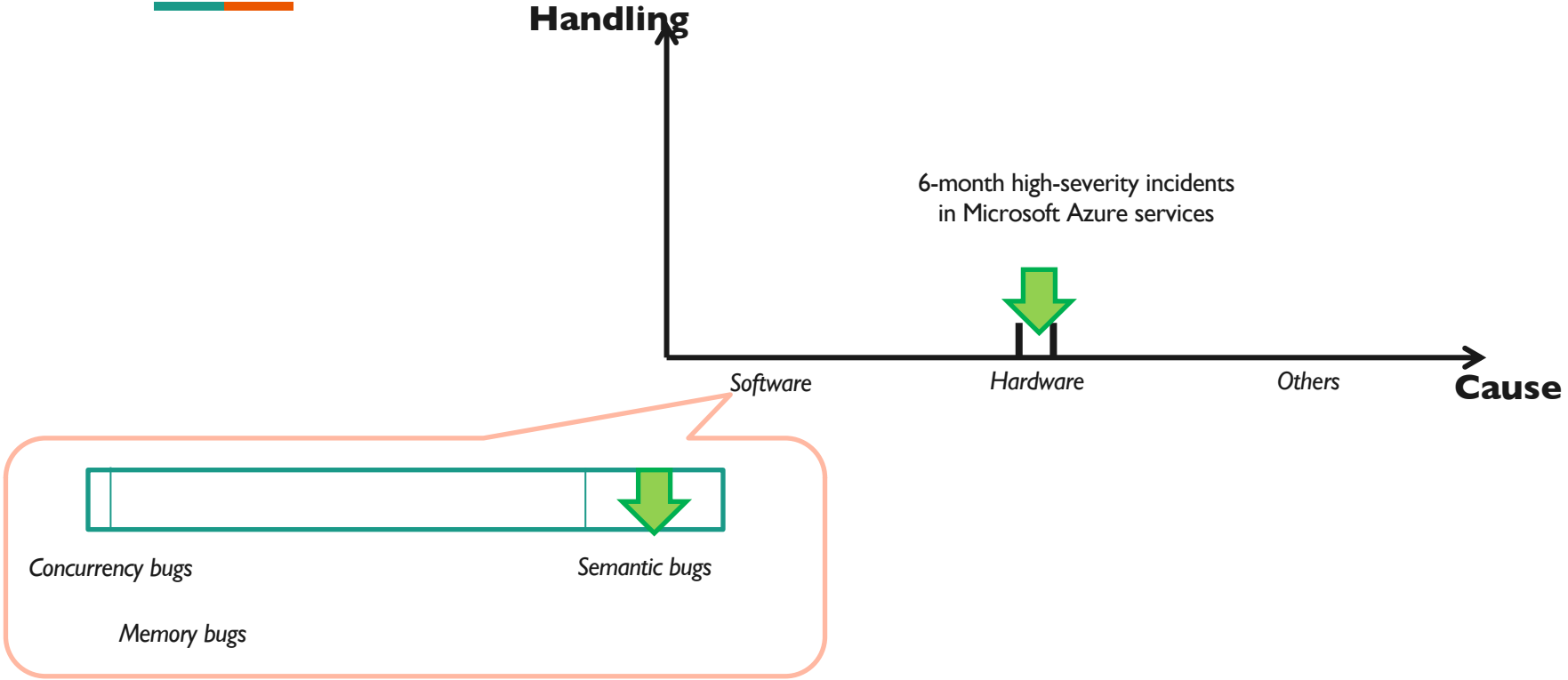
One more background …



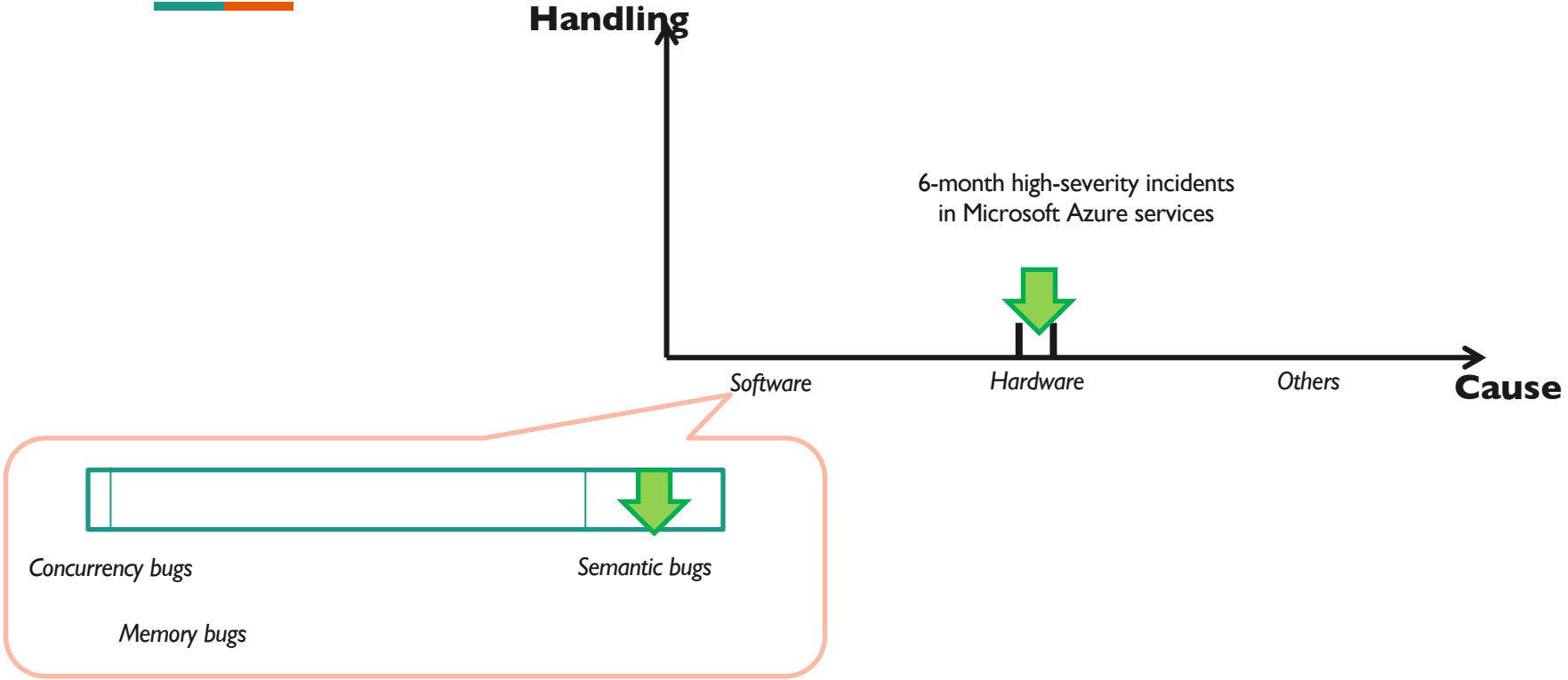
Our findings



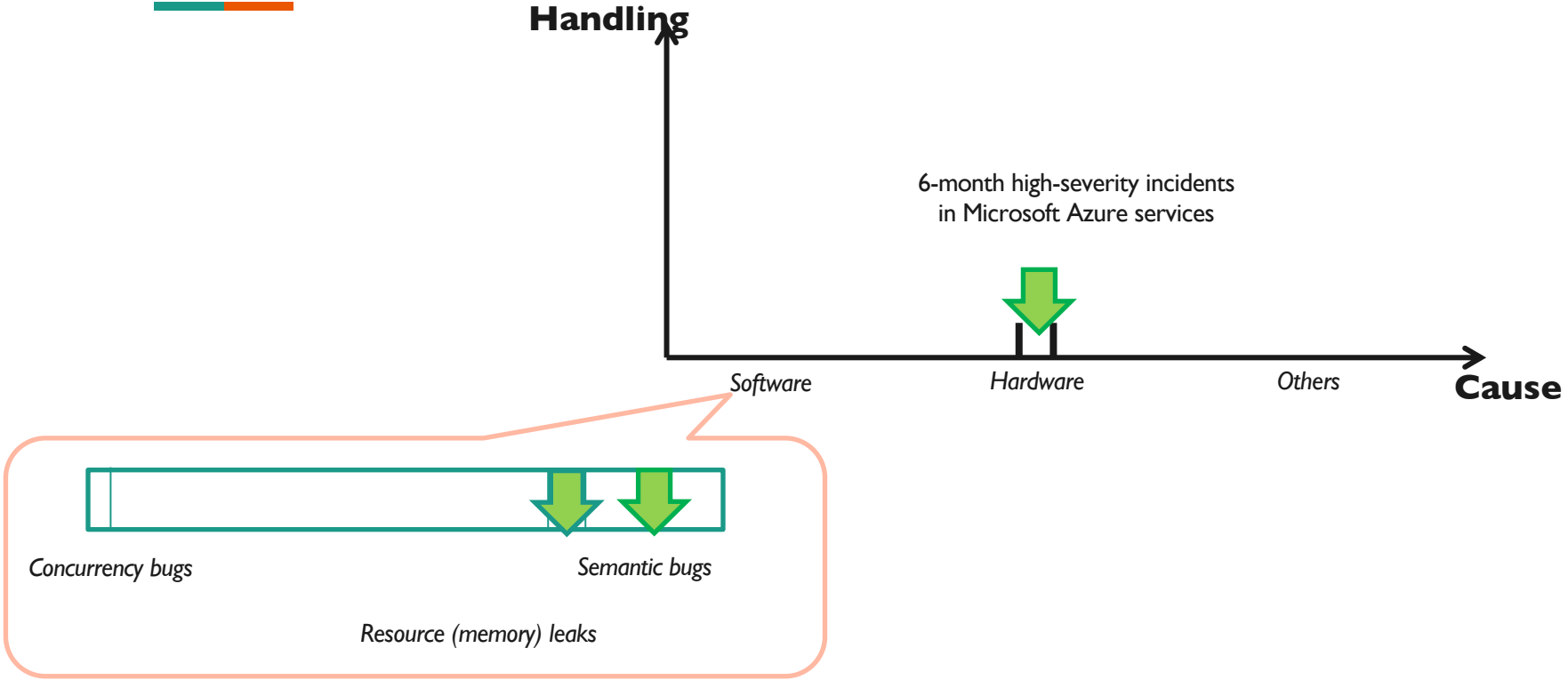
Our findings



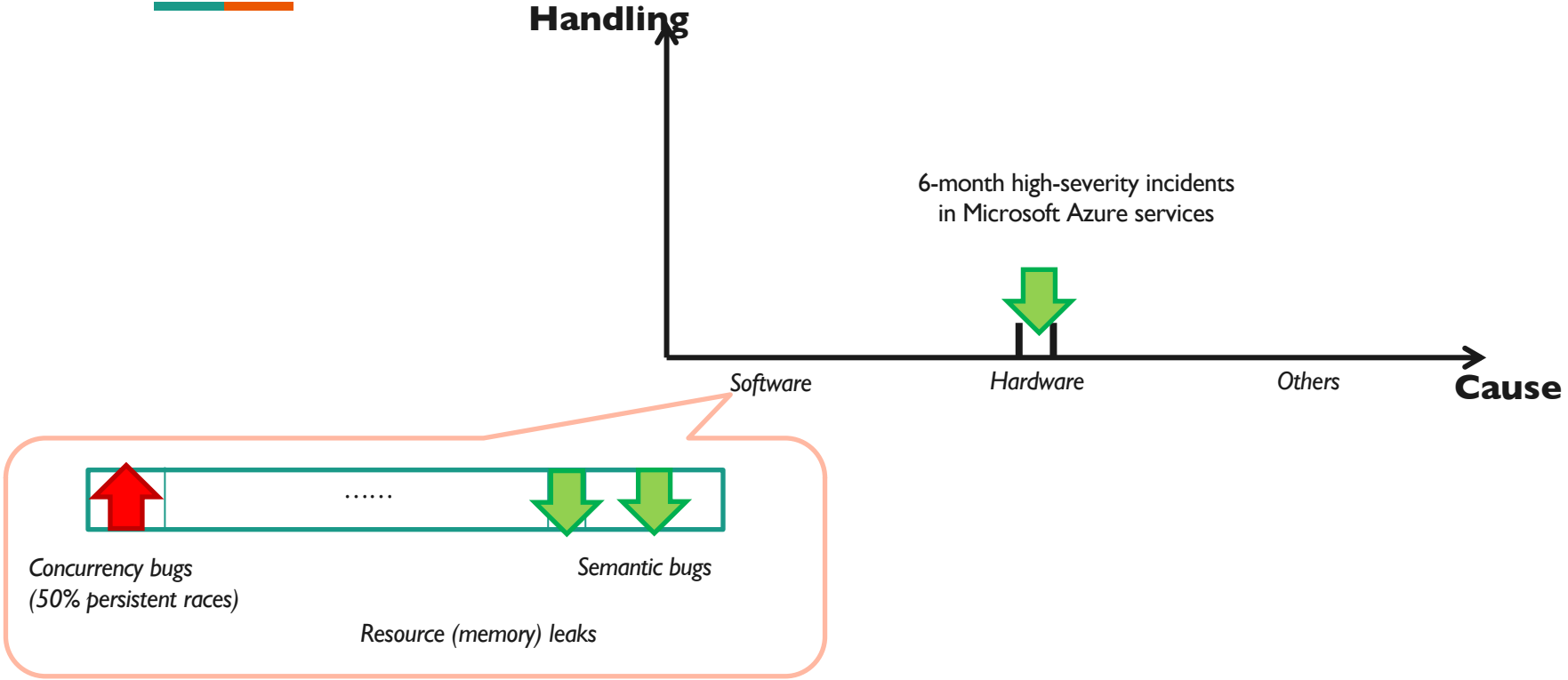
Our findings



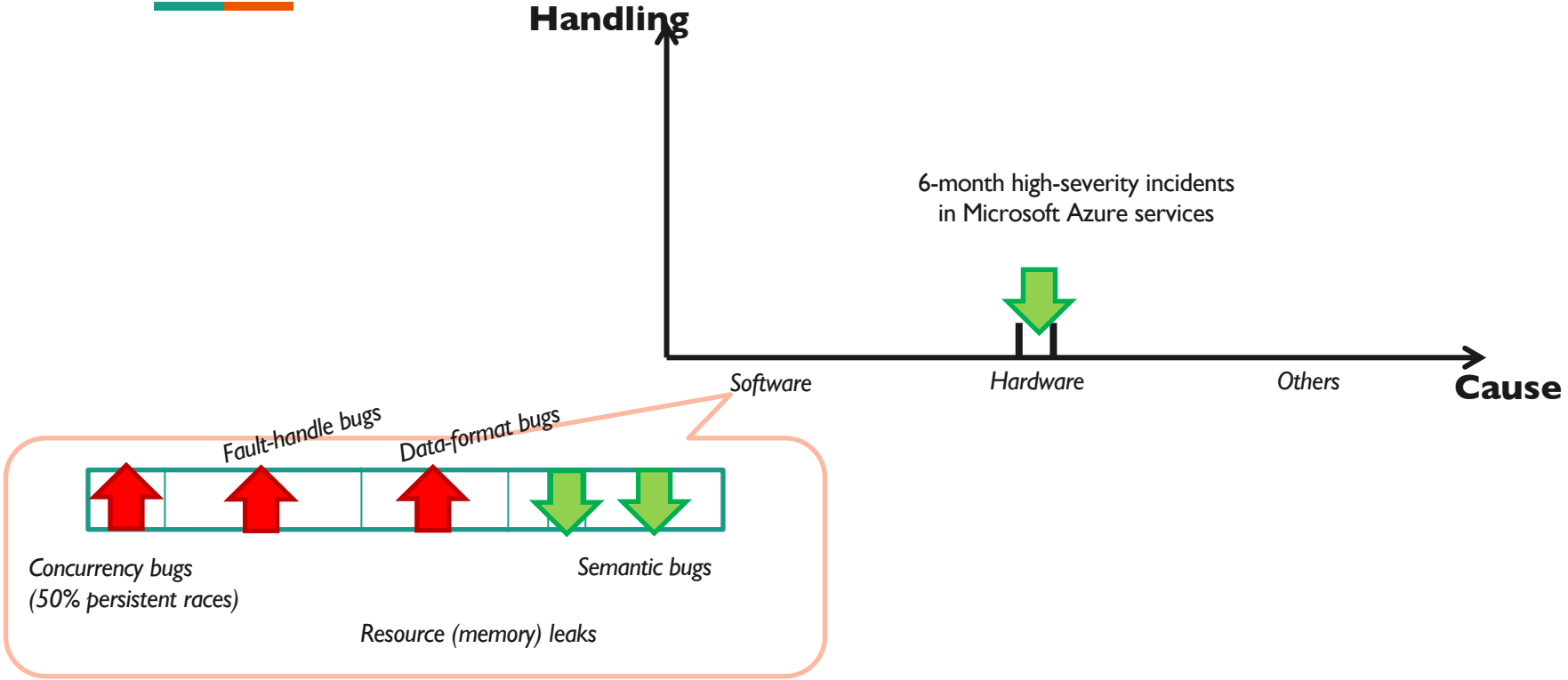
Our findings



Our findings

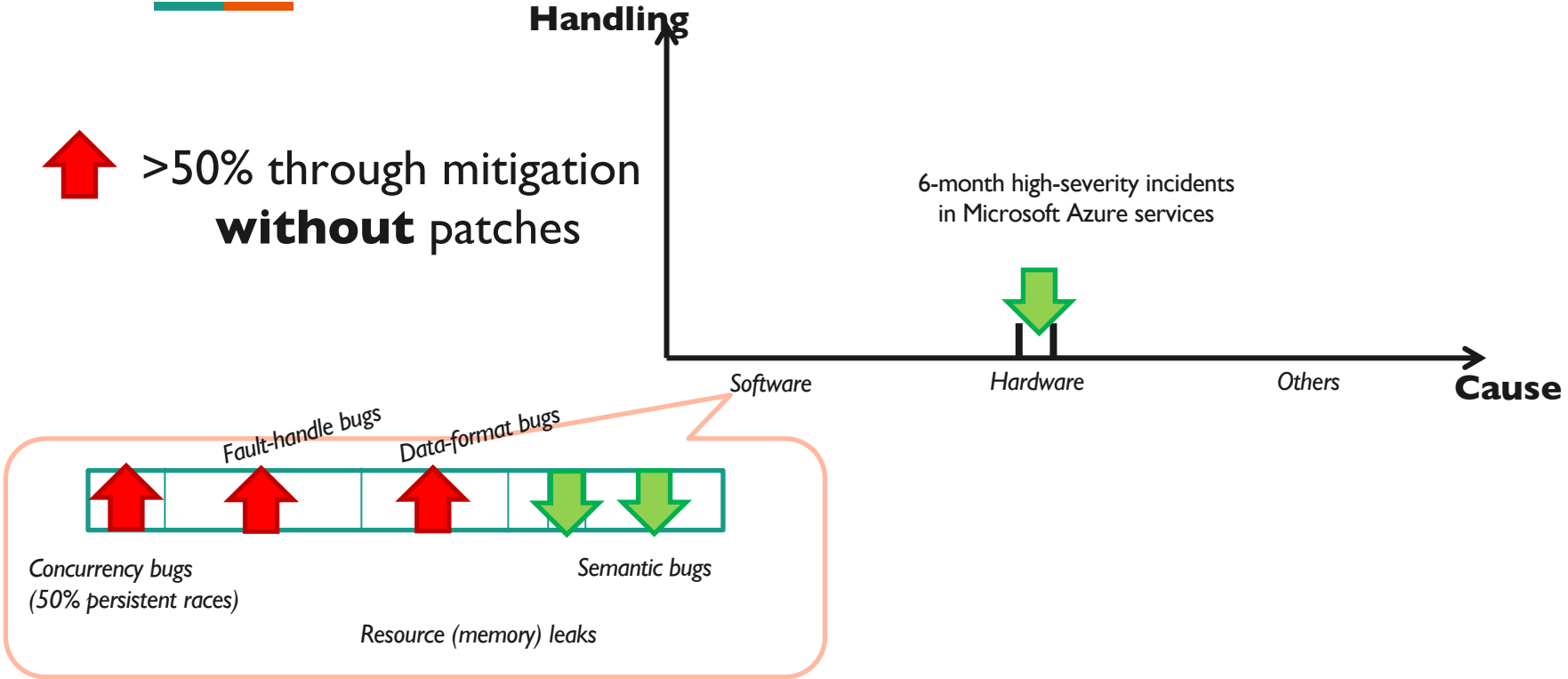


Our findings



Our findings

>50% through mitigation **without** patches

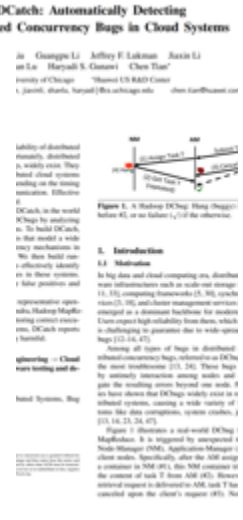


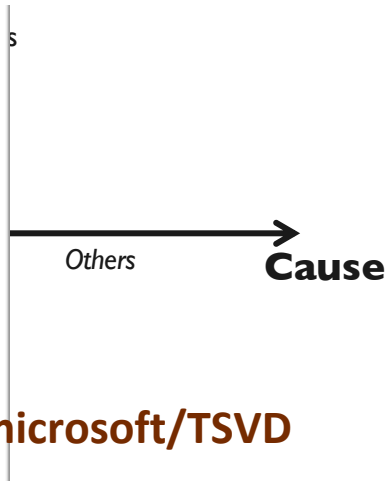
…n we do?

Handling

…h mitigation

…: pa…





Others

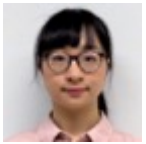
Cause

Concurrency bugs
(50% persistent races)

github.com/microsoft/TSVD

Conclusions

- Software bugs widely exist in big data & cloud systems
- Software bugs are taking on new forms in big data & cloud systems
  - Memory data ←→ Persistent data
- A lot of bug fighting can be done and to be done
- Our are making our bug set and tools open source!



Junwen Yang

Guangpu Li

UNIVERSITY of WASHINGTON

Microsoft

Thanks!