Rapid and Robust Impact Assessment of Software Changes in Large Internet-based Services

Shenglin Zhang, Ying Liu, Dan Pei

Yu Chen, Xianping Qu, Shimin Tao, Zhi Zang

清華大學 Tsinghua University

Baidu 百度

Internet-based Services

● Search	Google	bing	Baidu 百度
● Shopping	amazon	ebaY	淘宝网 Taobao.com
● Social	facebook	twitter	weibo
● Portal	YAHOO!	AOL	新浪网 sina.com
● Video	YouTube	blip	youku 优酷 世界都在看 土豆网 tudou.com

Software Change: Software Upgrade or Configuration Change

- Software upgrade

Introduce new feature

Improve performance

Fix bugs

Software Change: Software Upgrade or Configuration Change

- Software upgrade

Introduce new feature

Improve performance

Fix bugs

- Configuration change
  - e.g., traffic switching for load balancing reasons

Software Change: Software Upgrade or Configuration Change

- Software upgrade

Introduce new feature

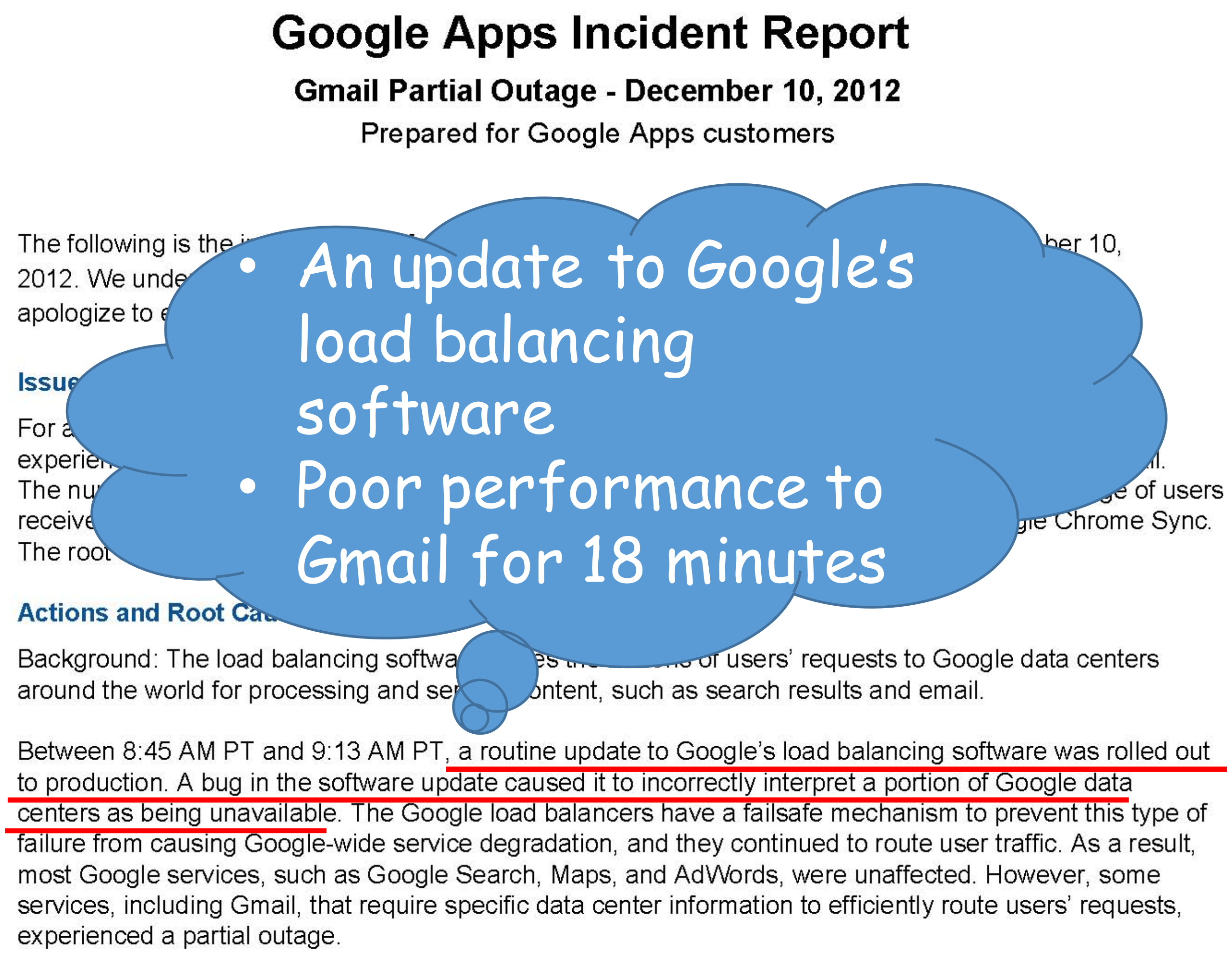
Improve performance

Fix bugs

- Configuration change
    - e.g., traffic switching for load balancing reasons
- Occurs frequently
    - 10K+ per day in Baidu

Impact of Erroneous Software Upgrades

2012.10, Google



Google Apps Incident Report

Gmail Partial Outage - December 10, 2012

Prepared for Google Apps customers

The following is the i ber 10, 2012. We unde apologize to e

Issue

For a experien l. The nu e of users receive gle Chrome Sync. The root

Actions and Root Ca

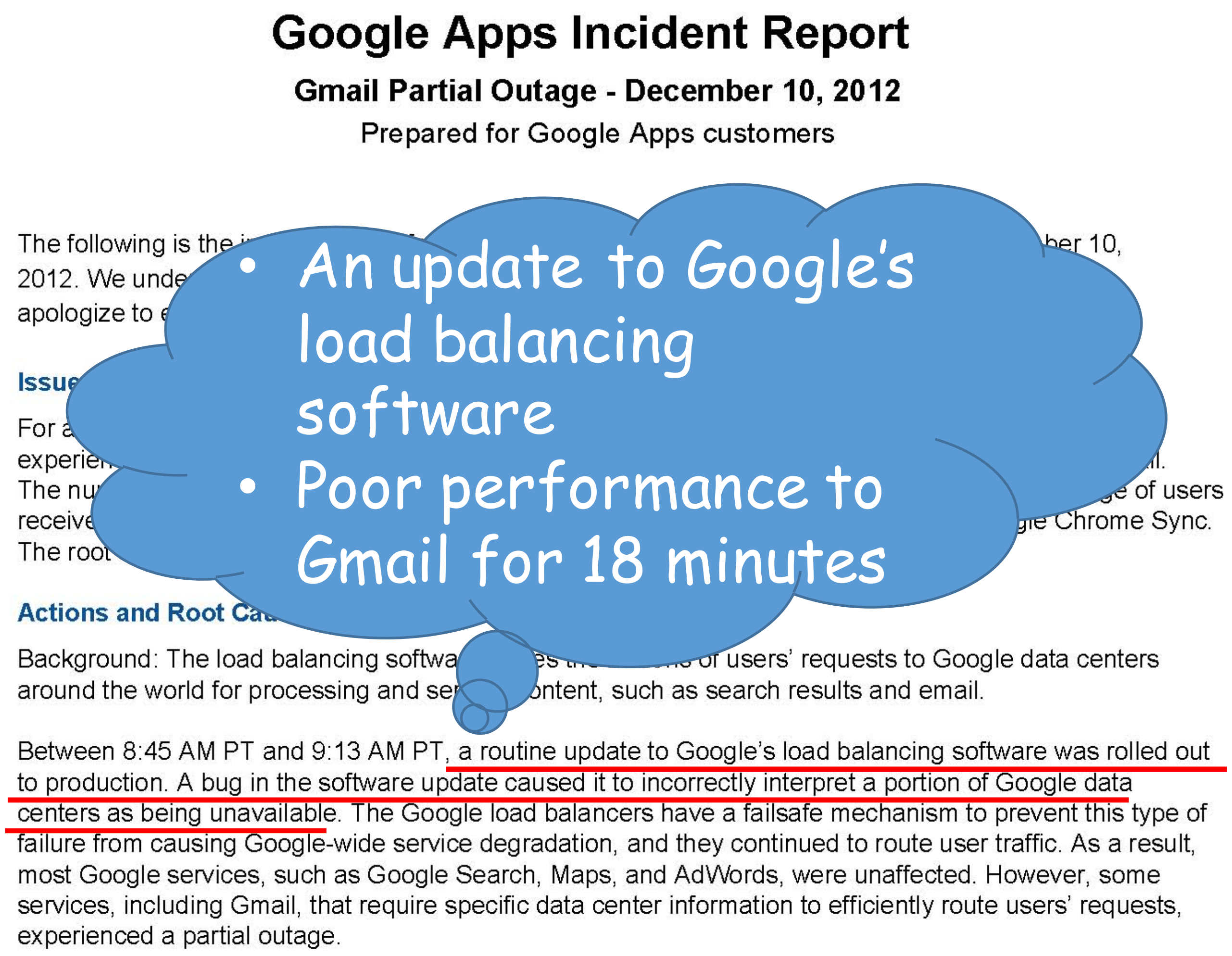
Background: The load balancing softwa es th of users' requests to Google data centers around the world for processing and se ontent, such as search results and email.

Between 8:45 AM PT and 9:13 AM PT, a routine update to Google's load balancing software was rolled out to production. A bug in the software update caused it to incorrectly interpret a portion of Google data centers as being unavailable. The Google load balancers have a failsafe mechanism to prevent this type of failure from causing Google-wide service degradation, and they continued to route user traffic. As a result, most Google services, such as Google Search, Maps, and AdWords, were unaffected. However, some services, including Gmail, that require specific data center information to efficiently route users' requests, experienced a partial outage.

- An update to Google's load balancing software
- Poor performance to Gmail for 18 minutes

Impact of Erroneous Software Upgrades

2012.10, Google



Google Apps Incident Report

Gmail Partial Outage - December 10, 2012

Prepared for Google Apps customers

The following is the i … ber 10,
2012. We unde …
apologize to e …

Issue

For a …
experien … il.
The nu … ge of users
receive … gle Chrome Sync.
The root …

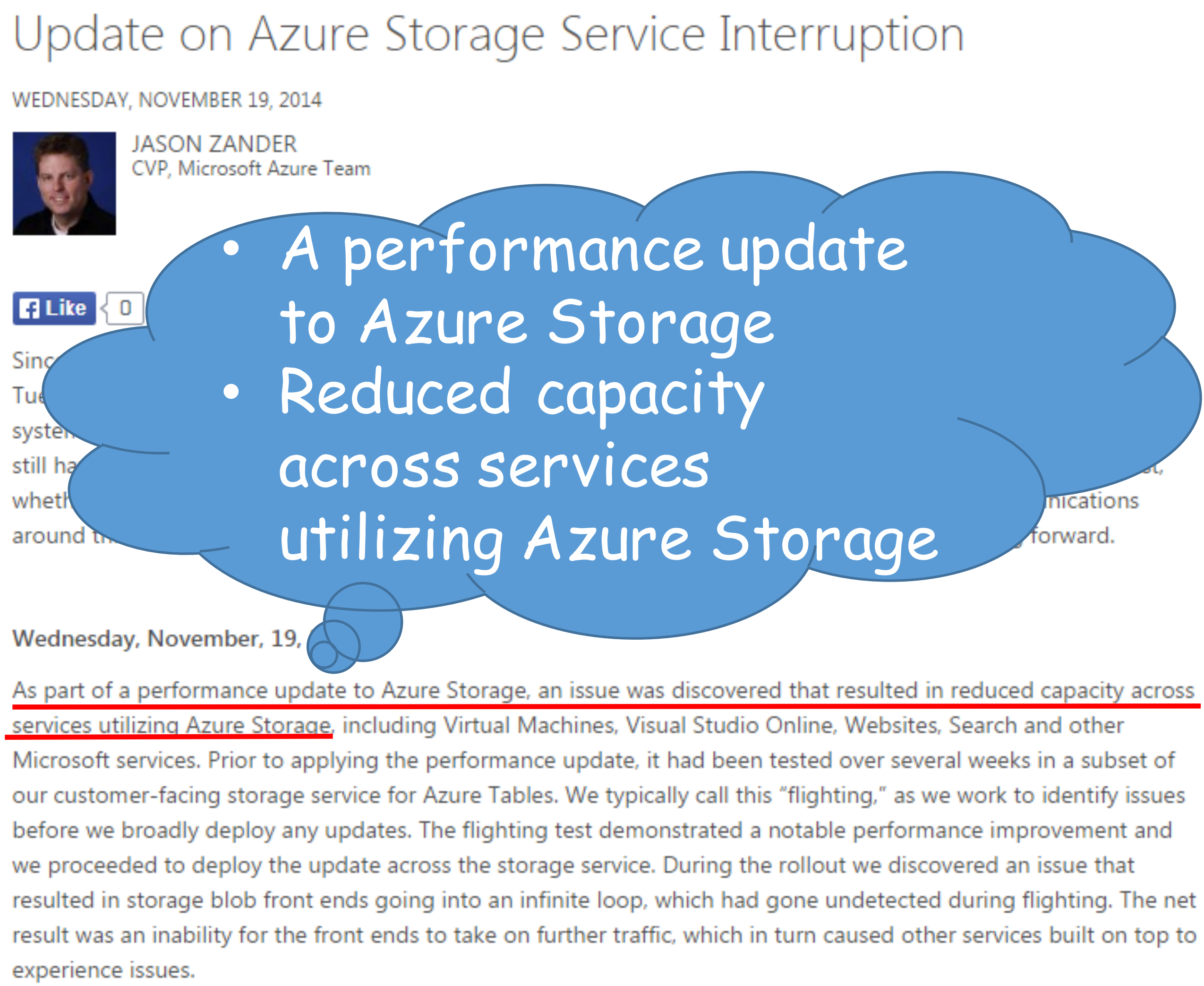
Actions and Root Ca

Background: The load balancing softwa … es t … of users' requests to Google data centers around the world for processing and se … ontent, such as search results and email.

Between 8:45 AM PT and 9:13 AM PT, a routine update to Google's load balancing software was rolled out to production. A bug in the software update caused it to incorrectly interpret a portion of Google data centers as being unavailable. The Google load balancers have a failsafe mechanism to prevent this type of failure from causing Google-wide service degradation, and they continued to route user traffic. As a result, most Google services, such as Google Search, Maps, and AdWords, were unaffected. However, some services, including Gmail, that require specific data center information to efficiently route users' requests, experienced a partial outage.

- An update to Google's load balancing software
- Poor performance to Gmail for 18 minutes

2014.11, Microsoft Azure



Update on Azure Storage Service Interruption

WEDNESDAY, NOVEMBER 19, 2014

JASON ZANDER
CVP, Microsoft Azure Team

Like 0

Sinc …
Tue …
syste …
still ha …
wheth … nications
around th … forward.

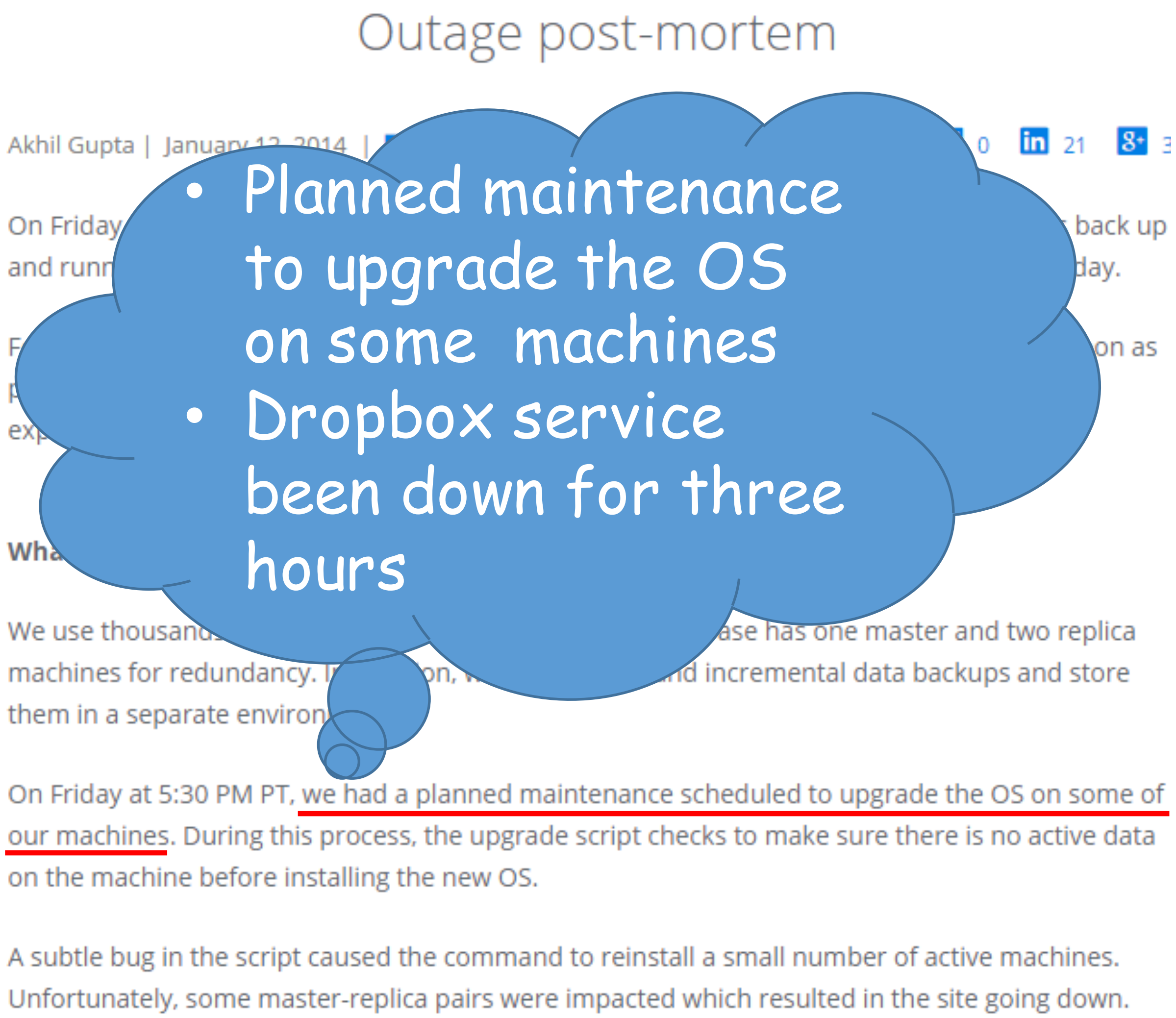
Wednesday, November, 19,

As part of a performance update to Azure Storage, an issue was discovered that resulted in reduced capacity across services utilizing Azure Storage, including Virtual Machines, Visual Studio Online, Websites, Search and other Microsoft services. Prior to applying the performance update, it had been tested over several weeks in a subset of our customer-facing storage service for Azure Tables. We typically call this "flighting," as we work to identify issues before we broadly deploy any updates. The flighting test demonstrated a notable performance improvement and we proceeded to deploy the update across the storage service. During the rollout we discovered an issue that resulted in storage blob front ends going into an infinite loop, which had gone undetected during flighting. The net result was an inability for the front ends to take on further traffic, which in turn caused other services built on top to experience issues.

- A performance update to Azure Storage
- Reduced capacity across services utilizing Azure Storage

Impact of Erroneous Configuration Changes

2014.1, Dropbox



Outage post-mortem

Akhil Gupta | January 12, 2014 | 0 in 21 3

On Friday … back up
and runn… day.

F… on as
p…
ex…

Wha…

We use thousand… ase has one master and two replica
machines for redundancy. I… on, w… nd incremental data backups and store
them in a separate environ…

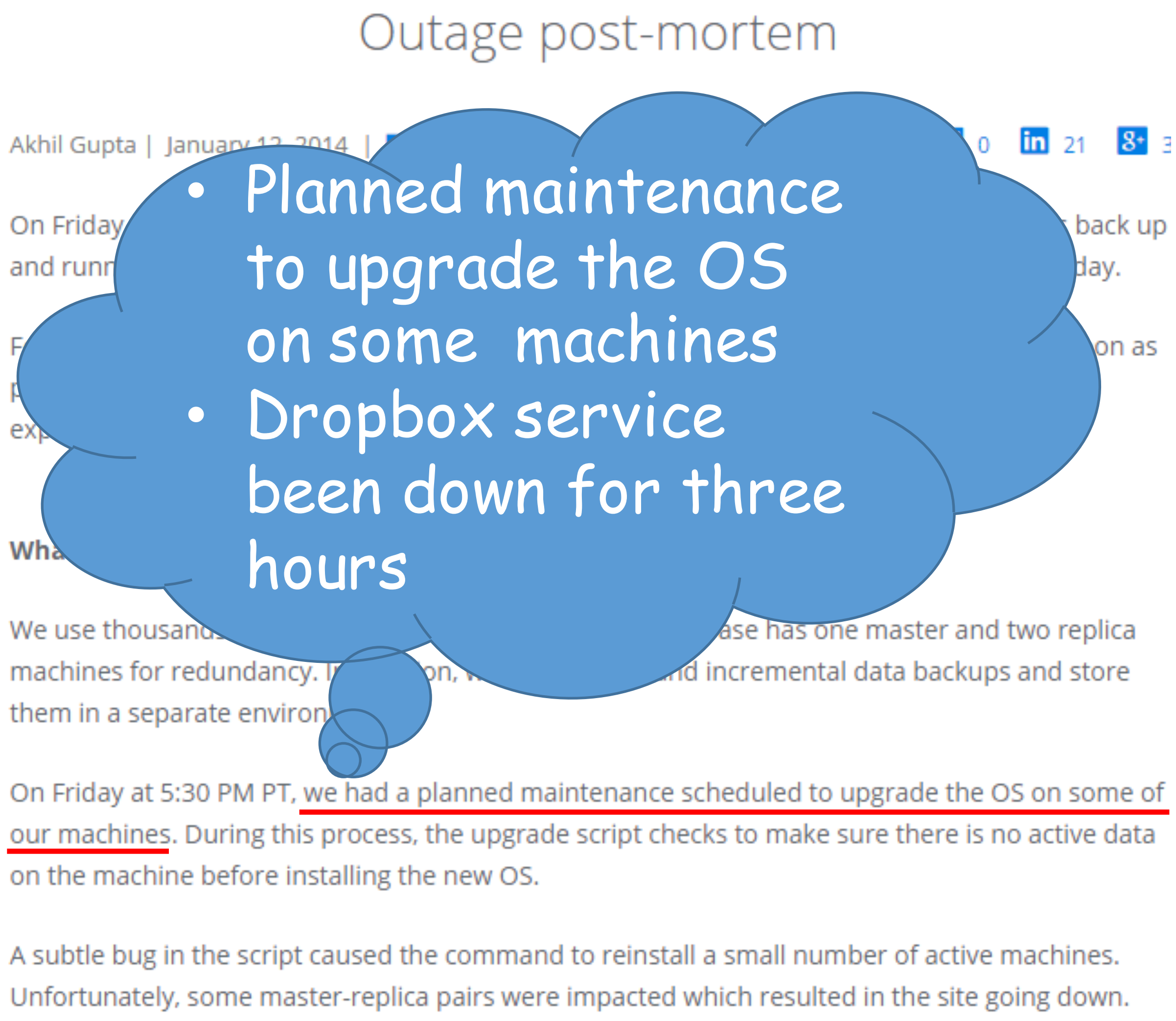
On Friday at 5:30 PM PT, we had a planned maintenance scheduled to upgrade the OS on some of our machines. During this process, the upgrade script checks to make sure there is no active data on the machine before installing the new OS.

A subtle bug in the script caused the command to reinstall a small number of active machines. Unfortunately, some master-replica pairs were impacted which resulted in the site going down.

- Planned maintenance to upgrade the OS on some machines
- Dropbox service been down for three hours

Impact of Erroneous Configuration Changes

2014.1, Dropbox



Outage post-mortem

Akhil Gupta | January 12, 2014 | 0 in 21 3

On Friday … back up

and runn… day.

F… on as

p…

exp…

Wha…

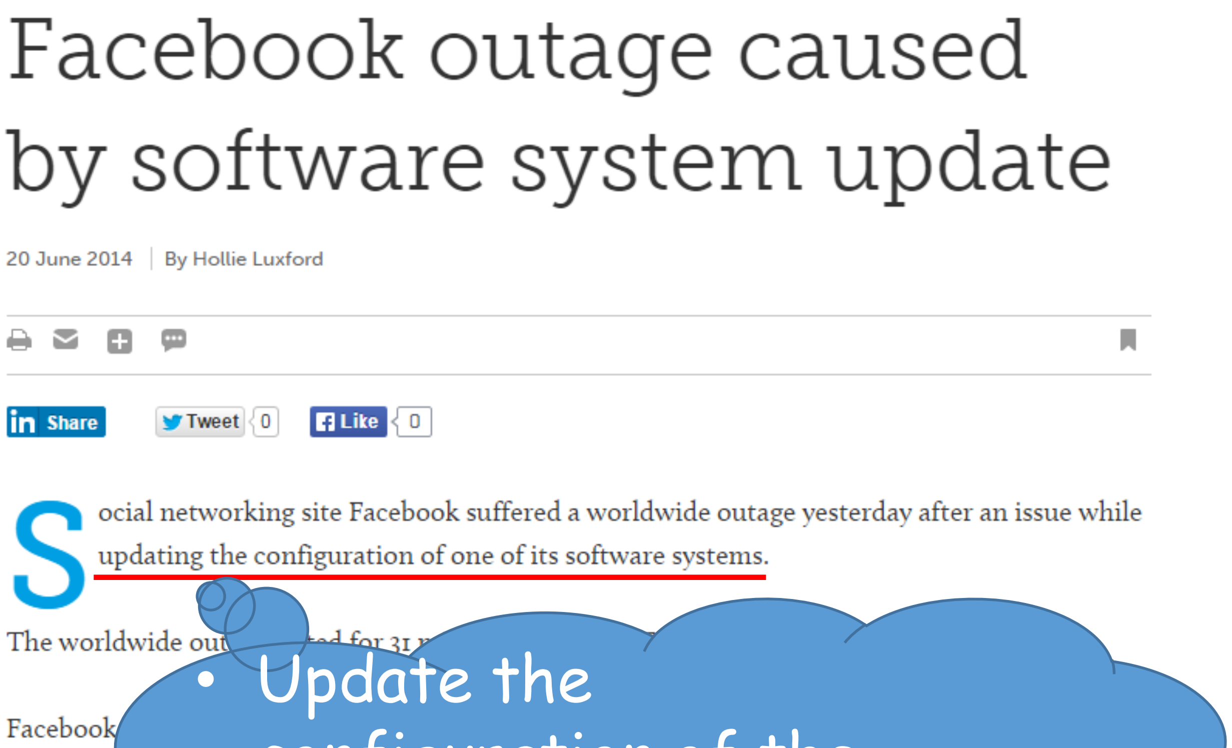
We use thousand… ase has one master and two replica machines for redundancy. I… on, w… nd incremental data backups and store them in a separate environ…

On Friday at 5:30 PM PT, we had a planned maintenance scheduled to upgrade the OS on some of our machines. During this process, the upgrade script checks to make sure there is no active data on the machine before installing the new OS.

A subtle bug in the script caused the command to reinstall a small number of active machines. Unfortunately, some master-replica pairs were impacted which resulted in the site going down.

- Planned maintenance to upgrade the OS on some machines
- Dropbox service been down for three hours

2014.6, Facebook



Facebook outage caused by software system update

20 June 2014 | By Hollie Luxford

Social networking site Facebook suffered a worldwide outage yesterday after an issue while updating the configuration of one of its software systems.

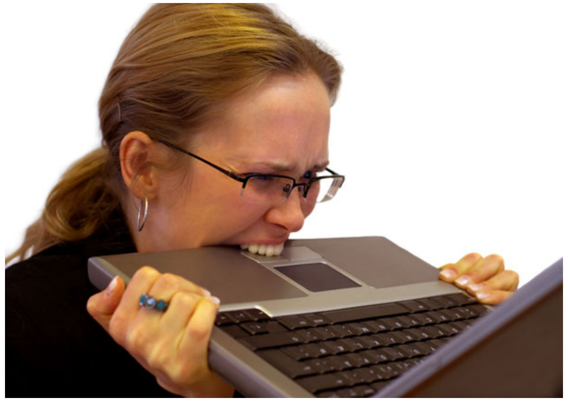
The worldwide out… d for 31 m…

Facebook…

- Update the configuration of the software systems
- Failed Facebook for 31 minutes

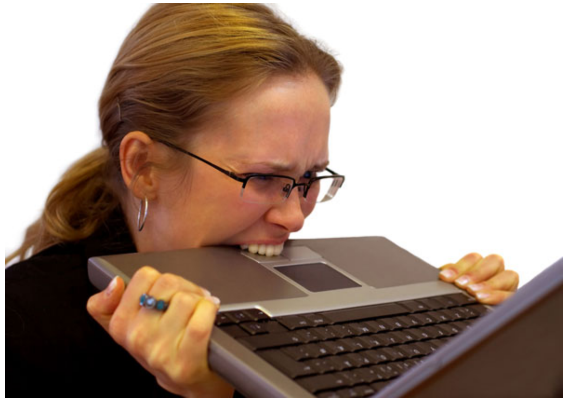
Impact of Erroneous Software Changes

- Poor user experience



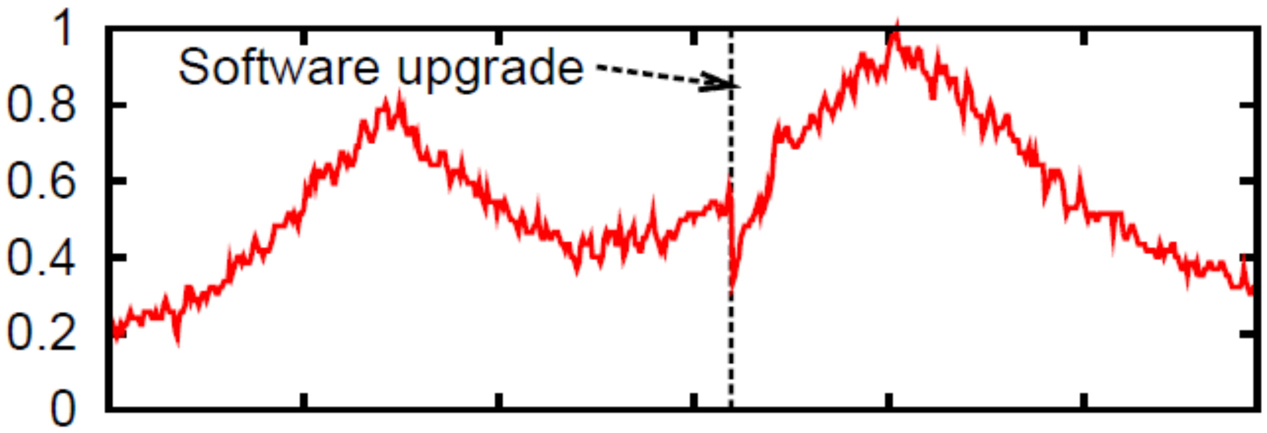
Impact of Erroneous Software Changes

- Poor user experience



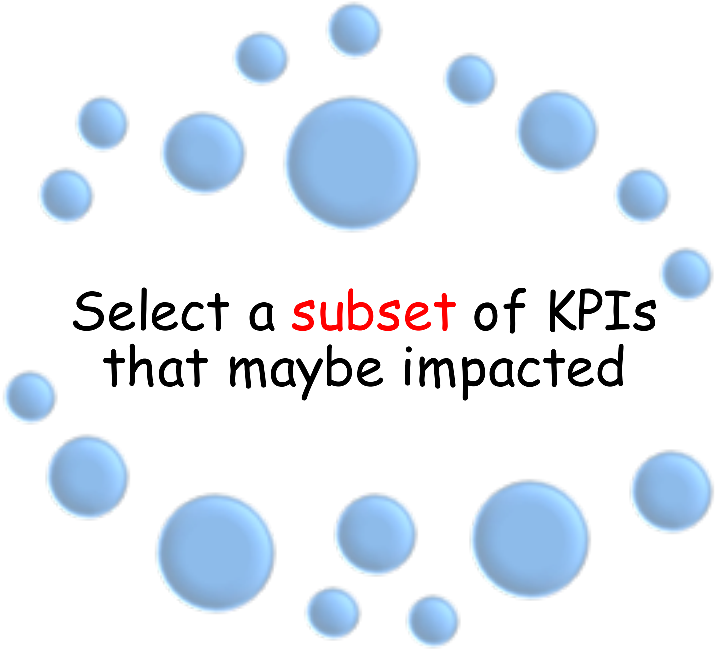
- A drop in revenue

The normalized number of successful orders



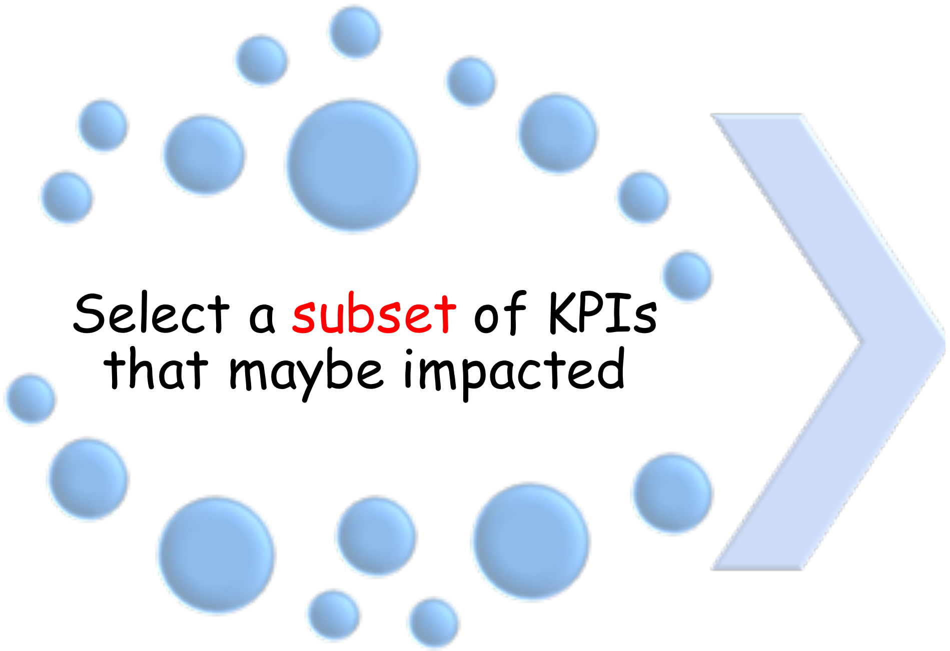
A real-world example

Manual Software Change Impact Assessment



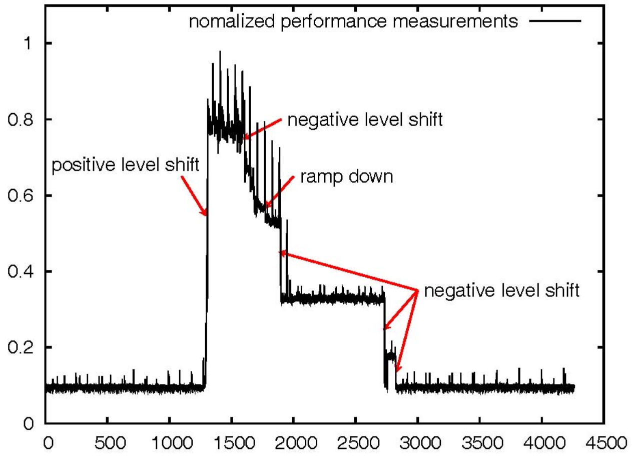
Select a subset of KPIs that maybe impacted

Manual Software Change Impact Assessment



Select a subset of KPIs that maybe impacted

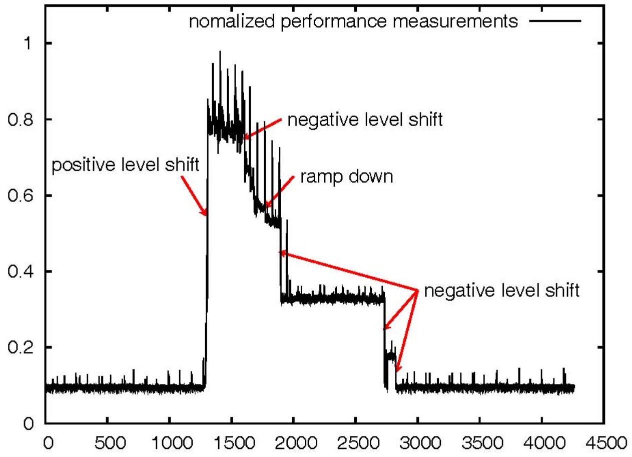
Inspect KPI changes



Manual Software Change Impact Assessment

Select a subset of KPIs that maybe impacted

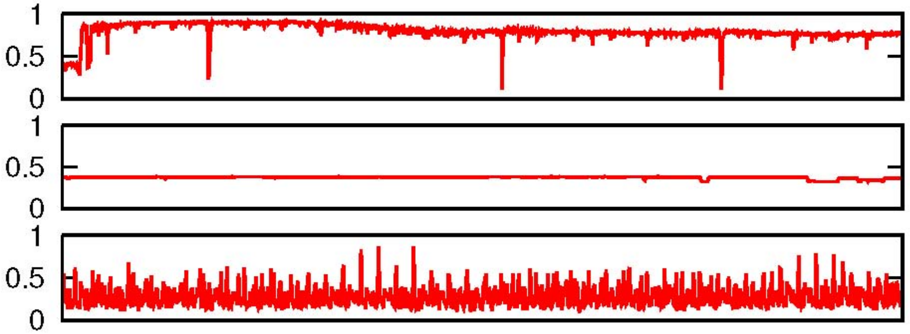
Inspect KPI changes



Decide whether to roll back

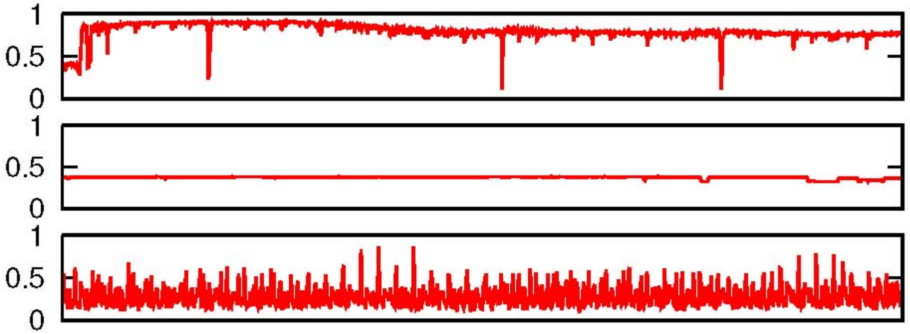
KPI (Key Performance Indicator) in Software Change

- KPIs of servers
  - CPU utilization
  - Memory utilization
  - NIC throughput
  - …

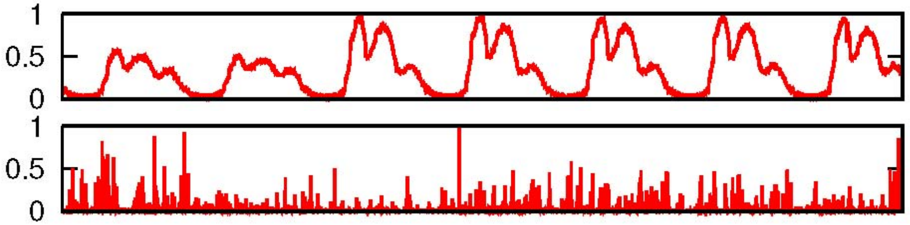


KPI (Key Performance Indicator) in Software Change

- KPIs of servers
  - CPU utilization
  - Memory utilization
  - NIC throughput
  - …

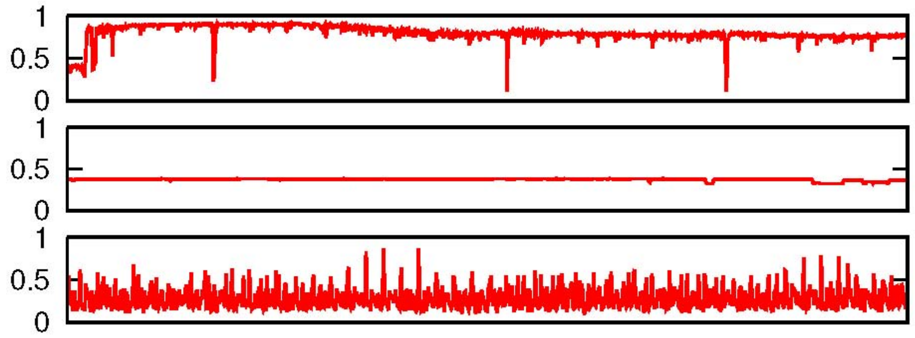


- KPIs of modules/processes
  - Web page view count
  - Web page view delay
  - …

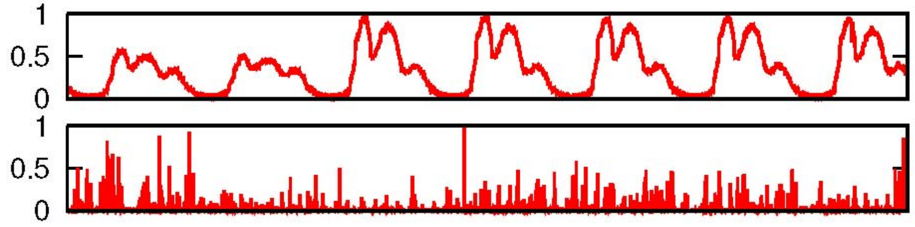


KPI (Key Performance Indicator) in Software Change

- KPIs of servers
  - CPU utilization
  - Memory utilization
  - NIC throughput
  - …



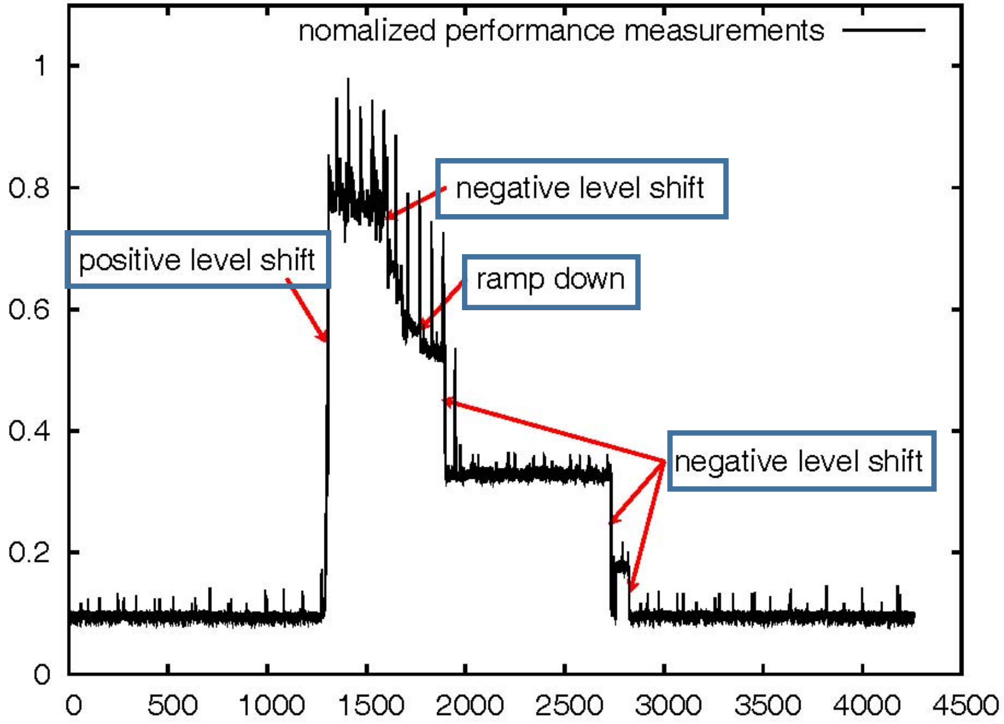
- KPIs of modules/processes
  - Web page view count
  - Web page view delay
  - …



- Up to hundreds of KPIs for a single software change

Definition of KPI Change: Level Shift or Ramp up/down

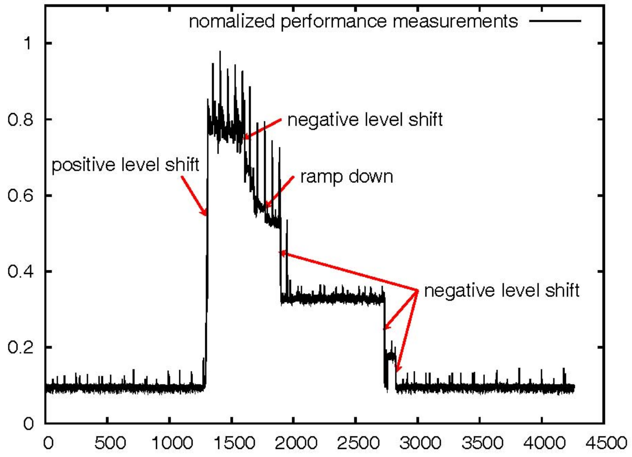
- KPI change
  - Indicative of performance increase/degradation
  - Hard to simulate in testbeds
  - Not reproducible



Manual Software Change Impact Assessment

Select a subset of KPIs that maybe impacted

Inspect KPI changes



Decide whether to roll back

- Labor-intensive
- Prone to error
- Not scalable

Design Goal

Software Change Impact Assessment System

Decide whether to roll back

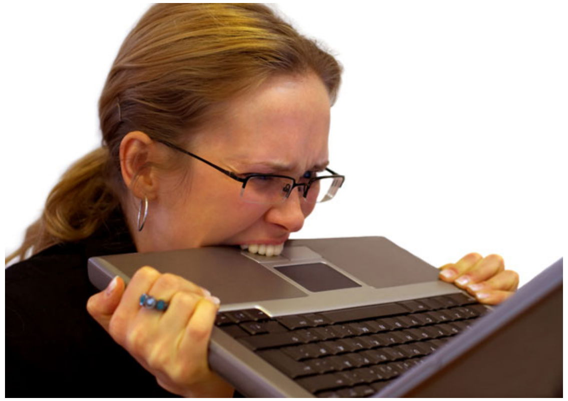
- Automatic
- Scalable
- Robust to various software changes and KPIs

Outline

- Background and Motivation
- Challenges
- Key Ideas
- Results
- Conclusion

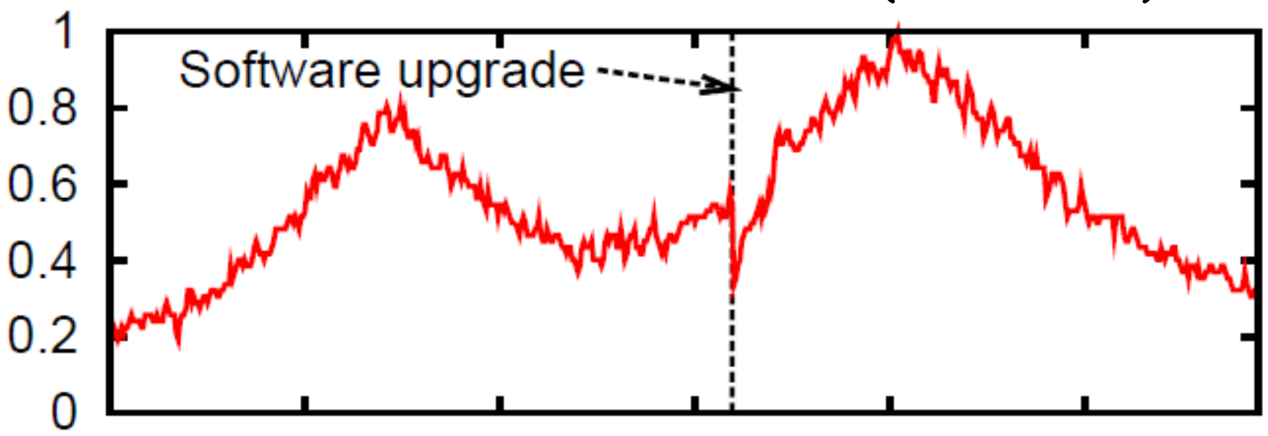
Challenge 1: Short Detection Delay Requirement Against Robustness

- Poor user experience



- A drop in revenue

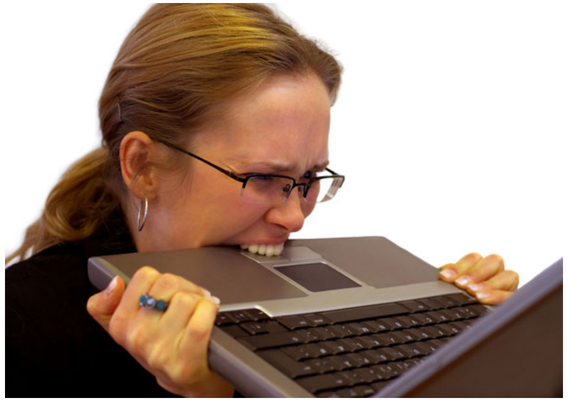
The number of successful orders (normalized)



A real-world example

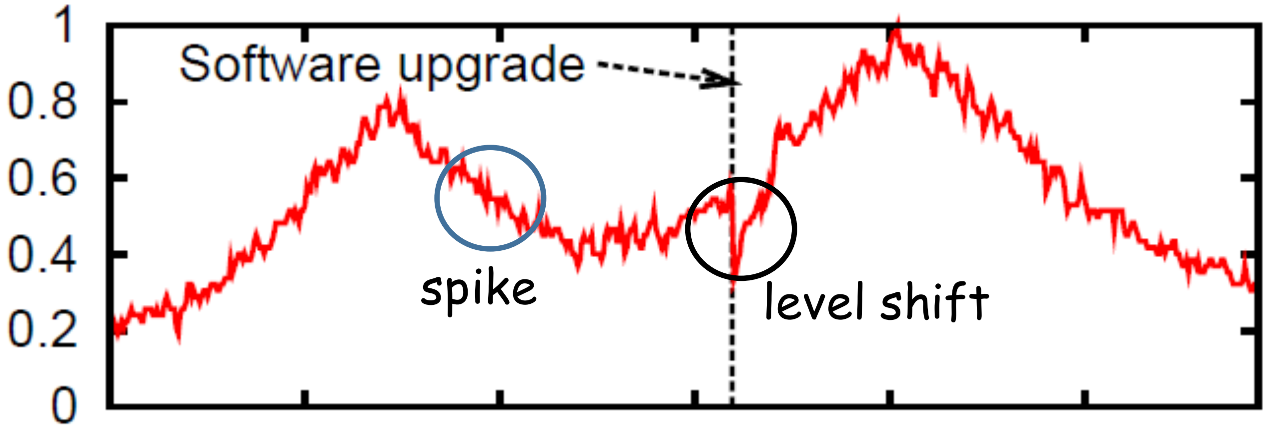
Challenge 1: Short Detection Delay Requirement Against Robustness

- Poor user experience



- A drop in revenue

The number of successful orders (normalized)



A real-world example

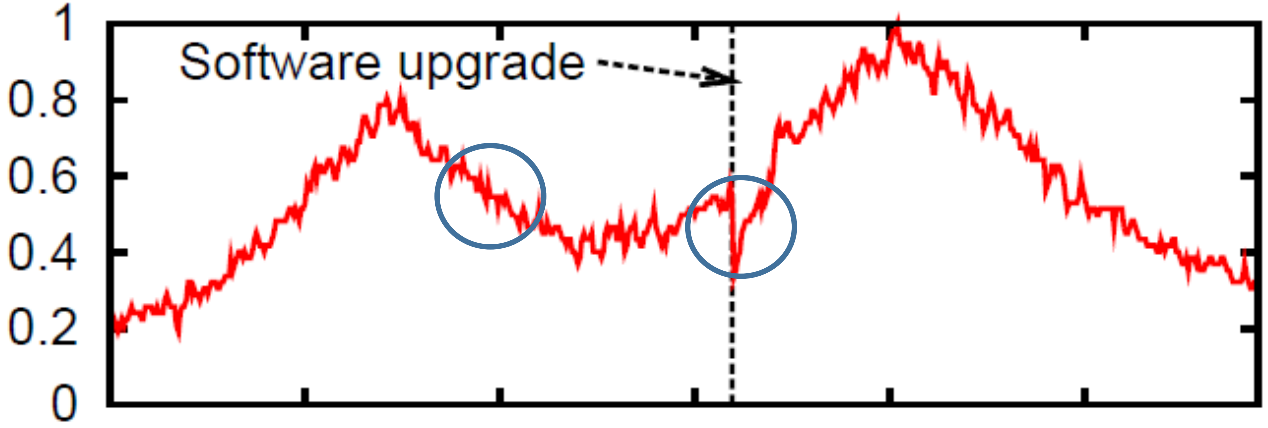
Challenge 1: Short Detection Delay Requirement Against Robustness

- Poor user experience

Detect KPI changes rapidly and accurately

- A drop in revenue

The number of successful orders (normalized)



A real-world example

Challenge 2: Large Number of KPIs

Challenge 2: Large Number of KPIs

100+ Internet-based services

20+ Internet-based services has 100+ million users

10k+ modules

500+ thousand servers

Challenge 2: Large Number of KPIs

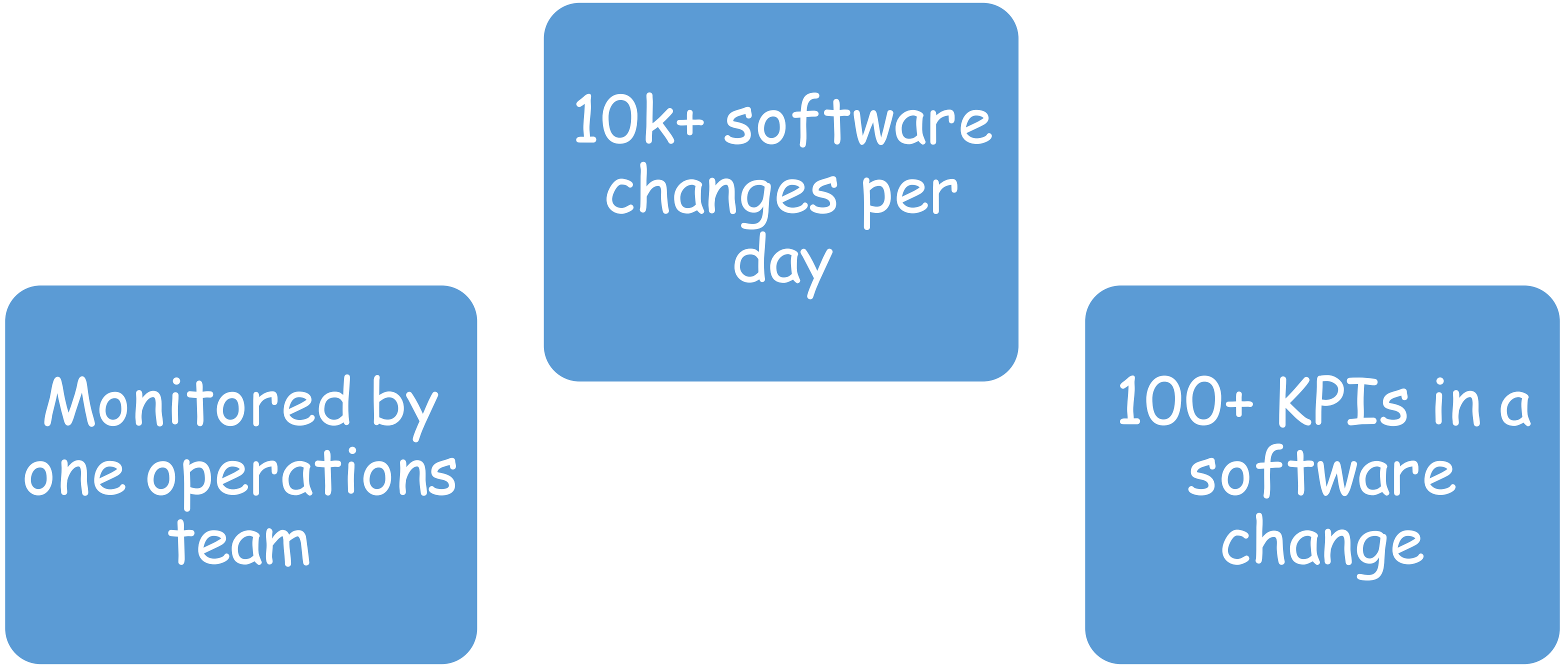
Monitored by one operations team

Challenge 2: Large Number of KPIs

10k+ software changes per day

Monitored by one operations team

Challenge 2: Large Number of KPIs

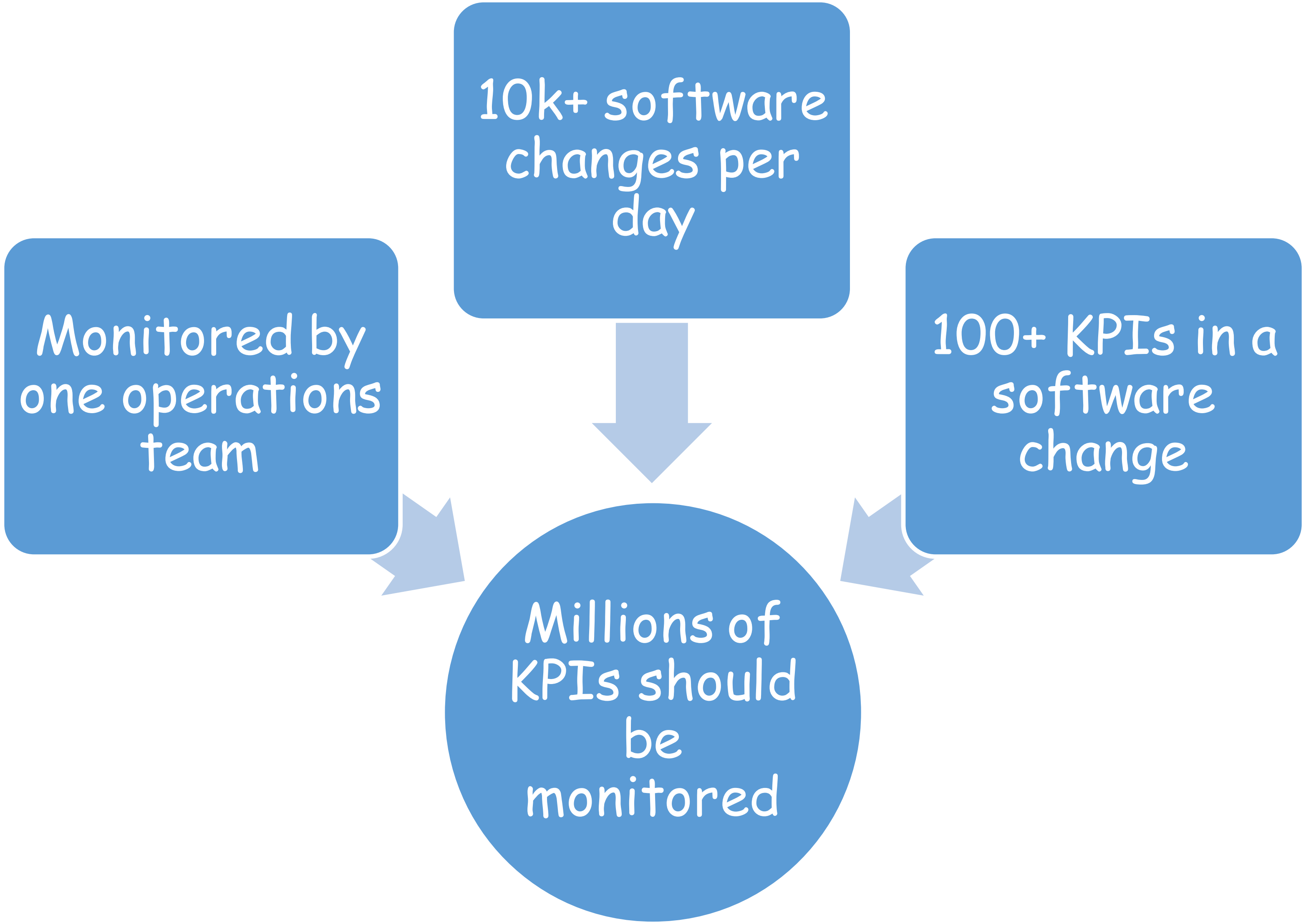


10k+ software changes per day

Monitored by one operations team

100+ KPIs in a software change

Challenge 2: Large Number of KPIs



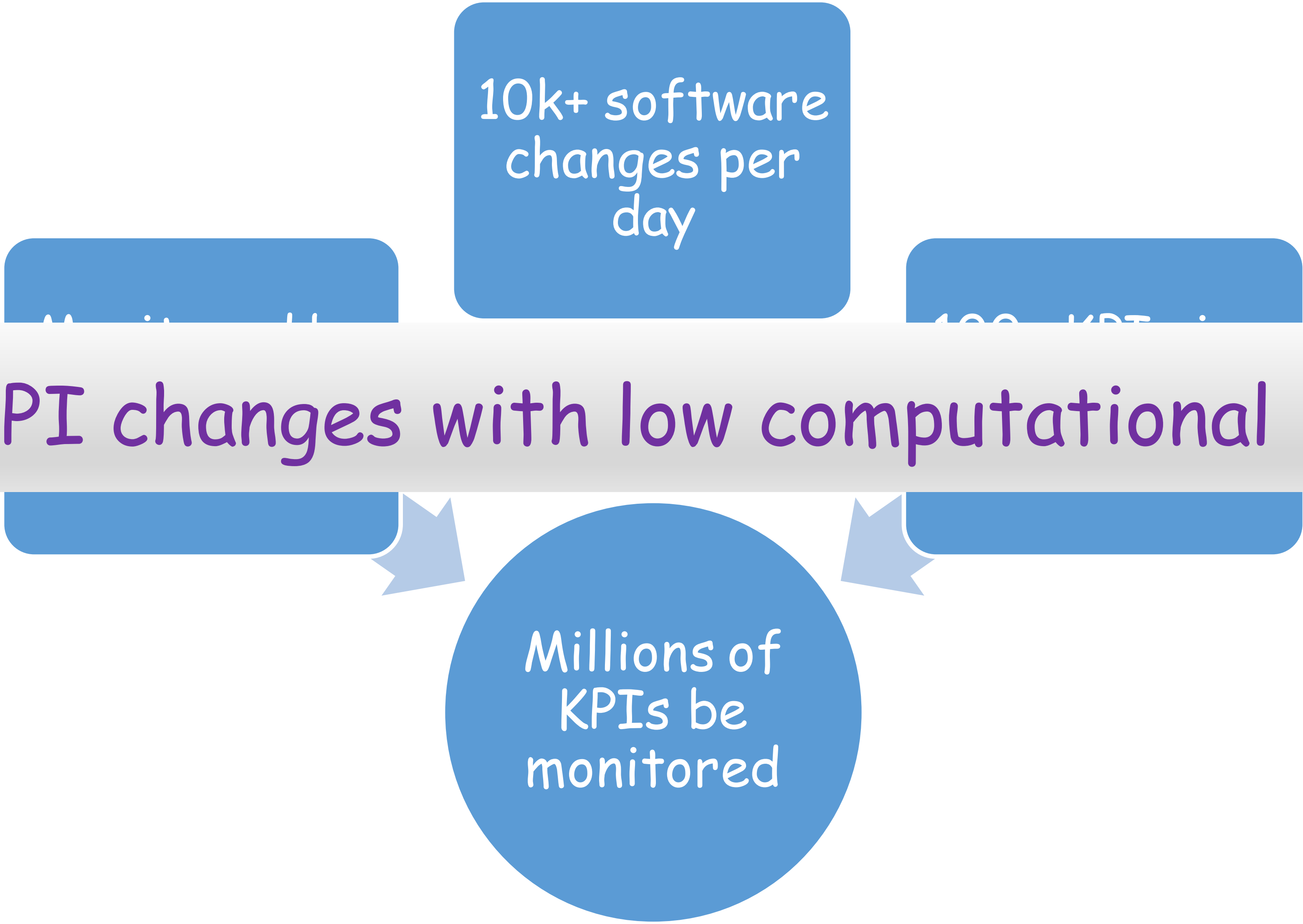
10k+ software changes per day

Monitored by one operations team

100+ KPIs in a software change

Millions of KPIs should be monitored

Challenge 2: Large Number of KPIs



10k+ software changes per day

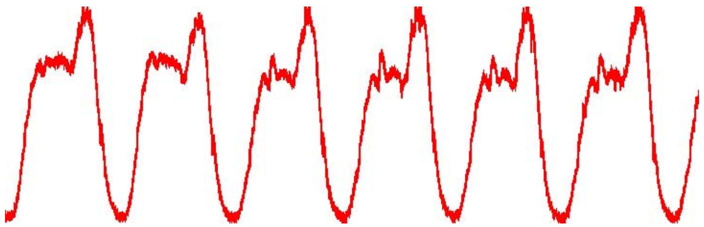
Millions of KPIs be monitored

Detect KPI changes with low computational cost

Challenge 3: Diverse Types of Data

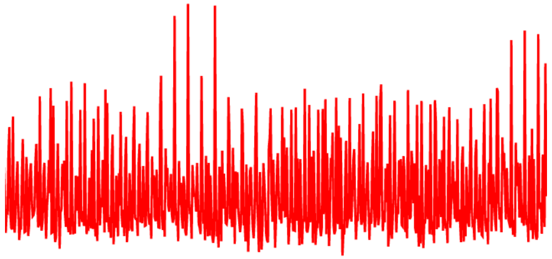
- Diverse types of KPI data

Seasonal



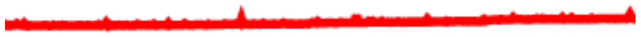
Page view count

Variable



NIC throughput

Stationary



Memory utilization

Challenge 3: Diverse Types of Data

- Diverse types of KPI data

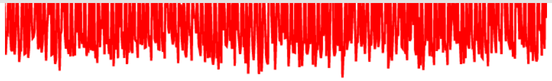
Seasonal

Variable

Stationary

Robust to various KPIs

Page view count



NIC throughput

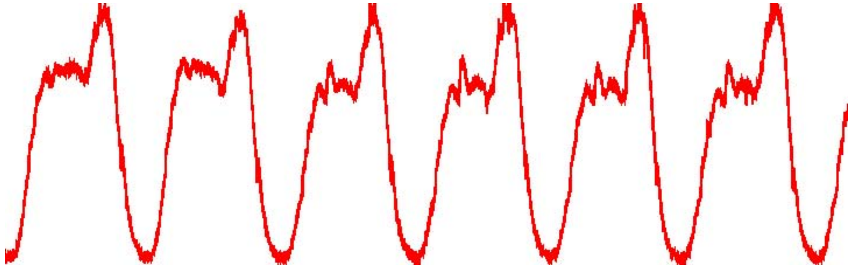
Memory utilization

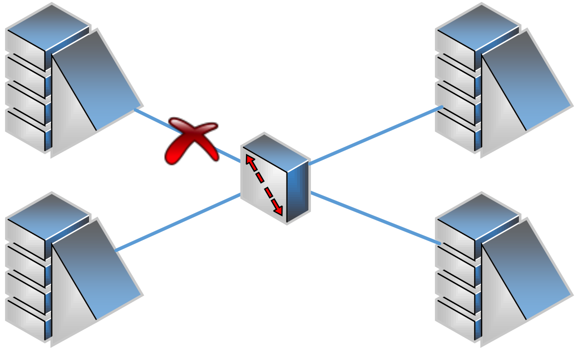
Challenge 4: KPI Changes Maybe Caused by Other Factors

Seasonality

Network breakdowns

Malicious attacks





Challenge 4: KPI Changes Maybe Caused by Other Factors

Seasonality

Network breakdowns

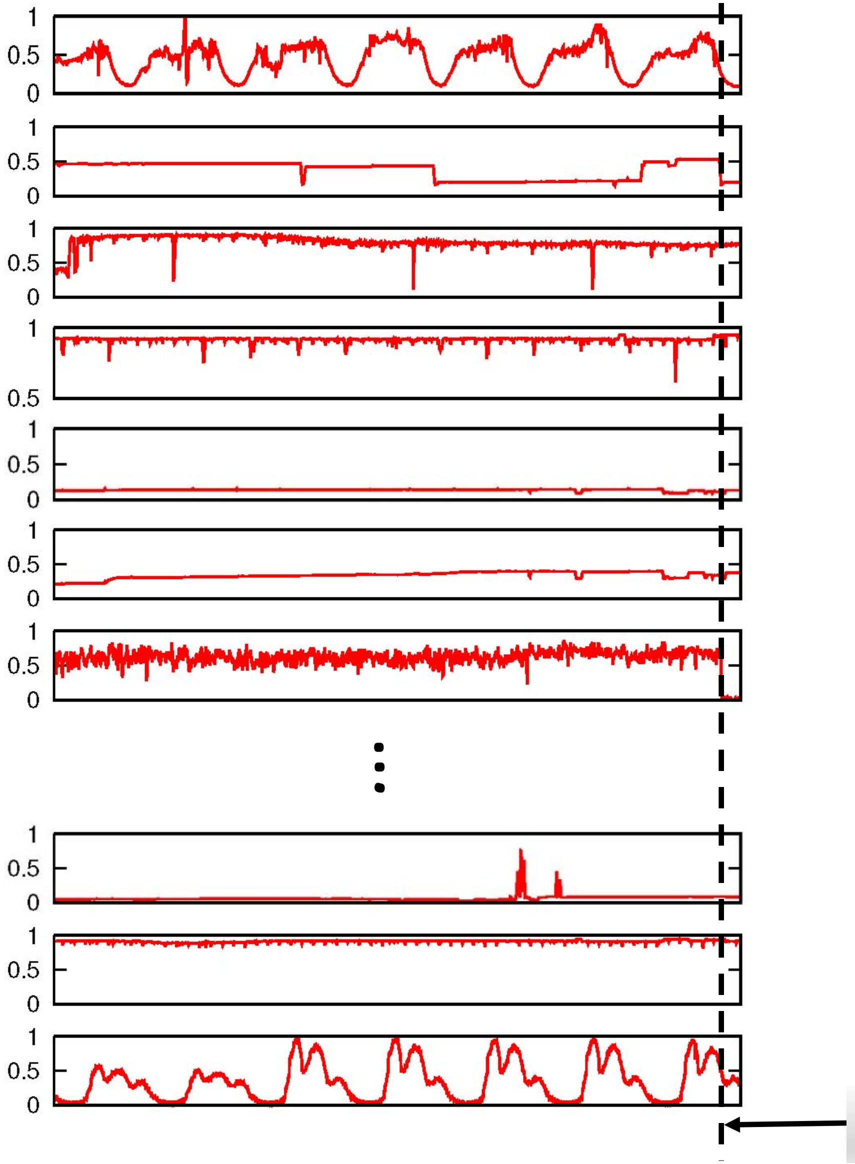
Malicious attacks

Eliminate KPI changes induced by other factors

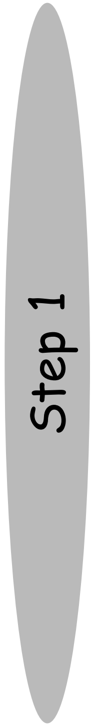
Outline

- Background and Motivation
- Challenges
- Key Ideas
- Results
- Conclusion

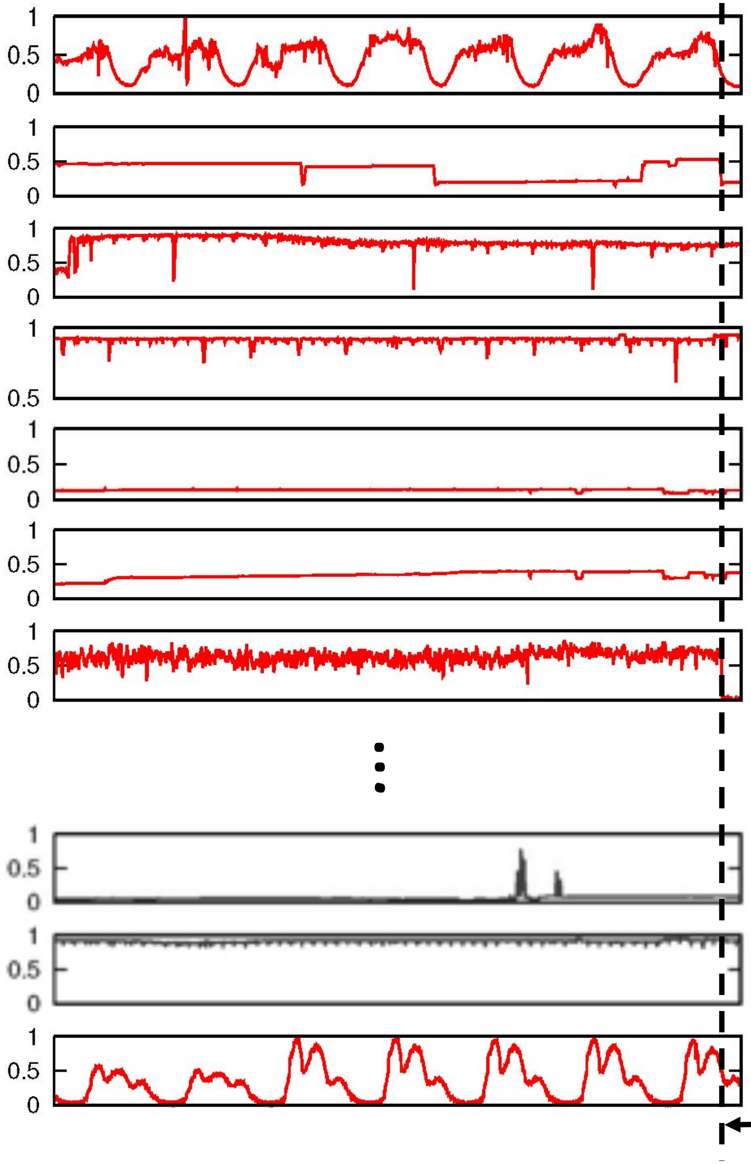
Design Overview



Step 1 – Identify the impact set



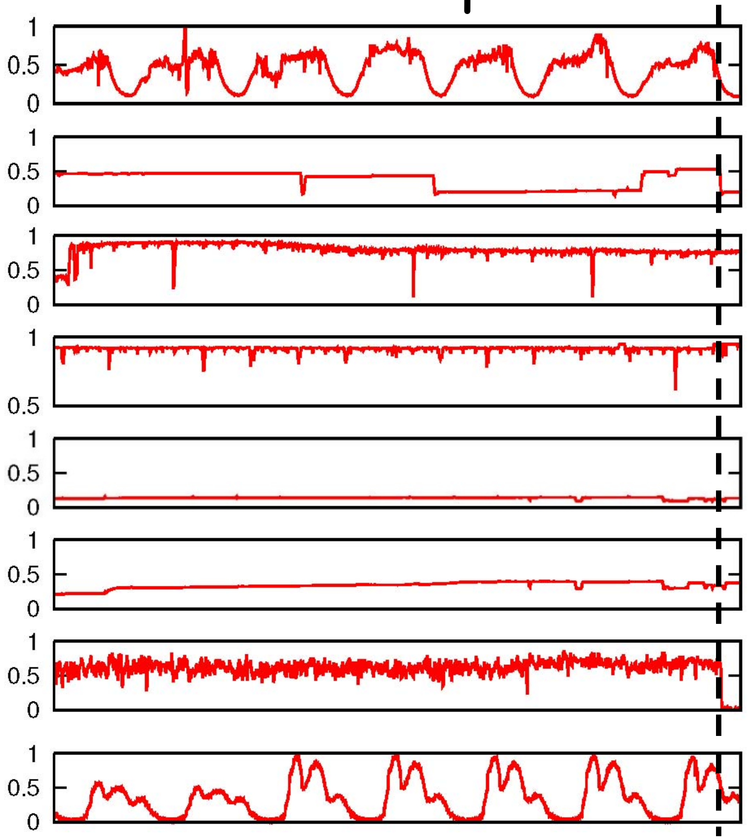
Design Overview



Step 1 – Identify the impact set

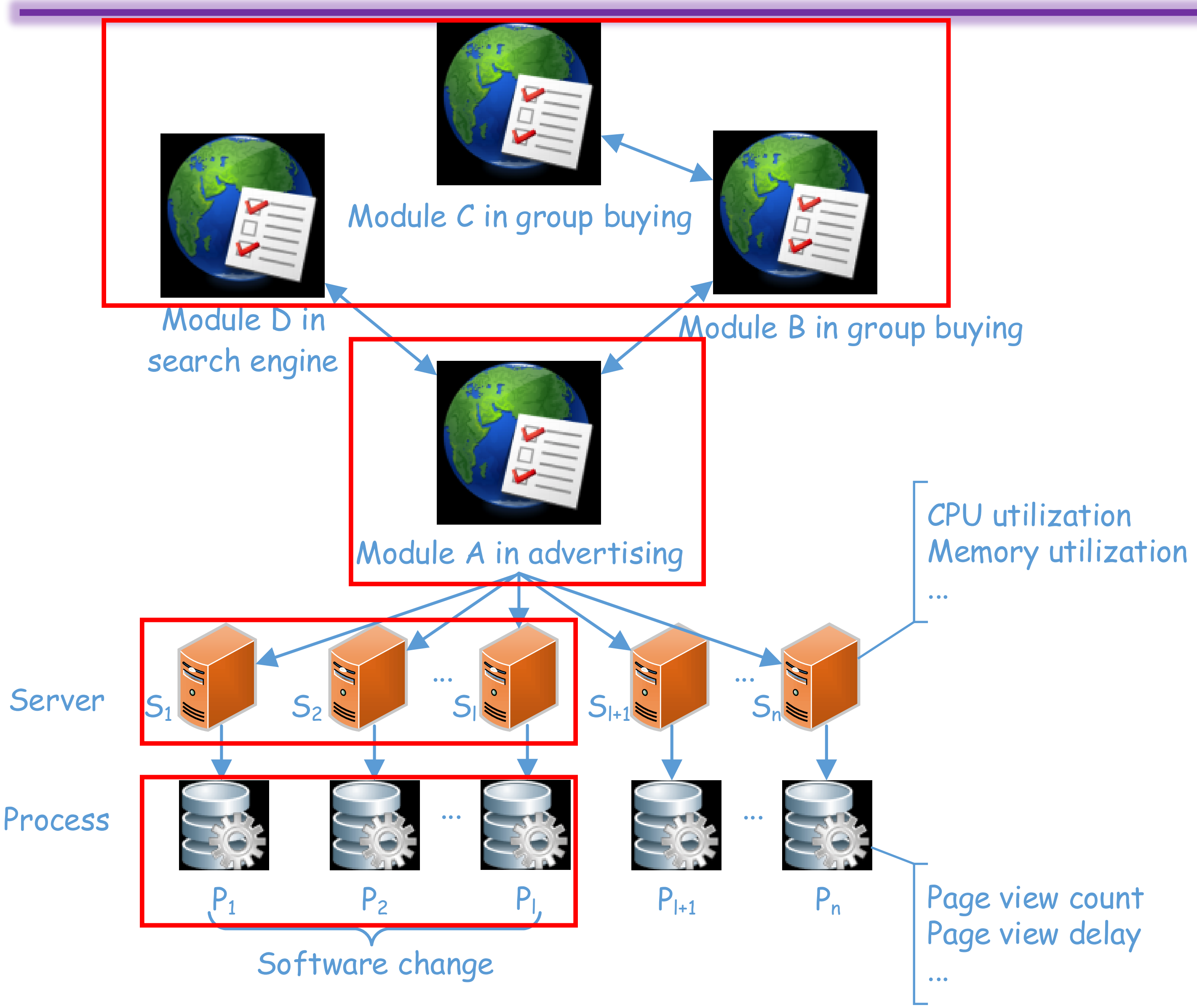
Step 1

KPIs in the impact set

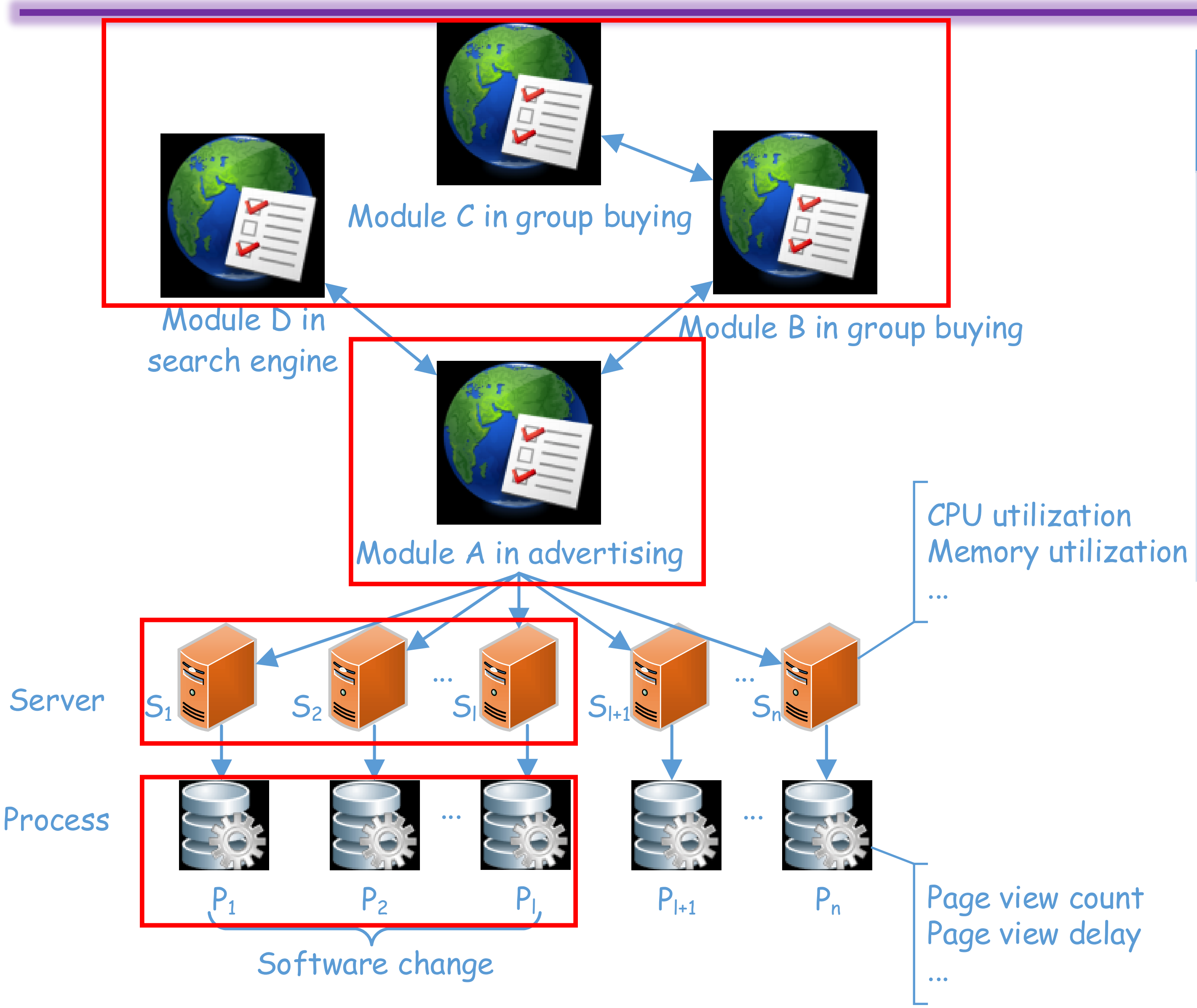


Software change in module A

Identify the Impact Set: Automatically Retrieve the Relevant KPIs



Identify the Impact Set: Automatically Retrieve the Relevant KPIs



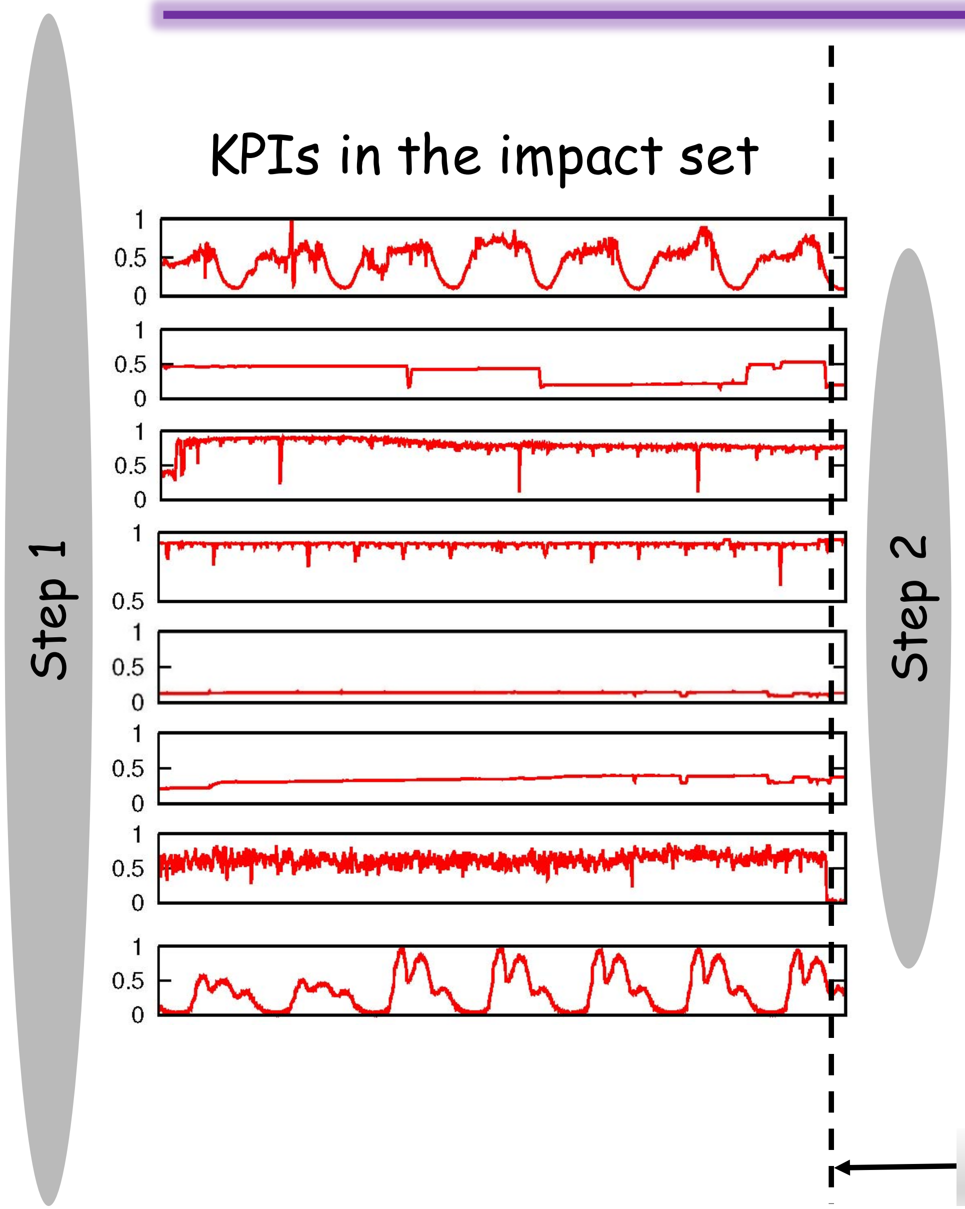
Input from operators

- Modules related module A: module B, C, D
- Servers/processes where the software change is deployed.

Design Overview

Step 1 – Identify the impact set

Step 2 – Detect behavior changes in KPIs

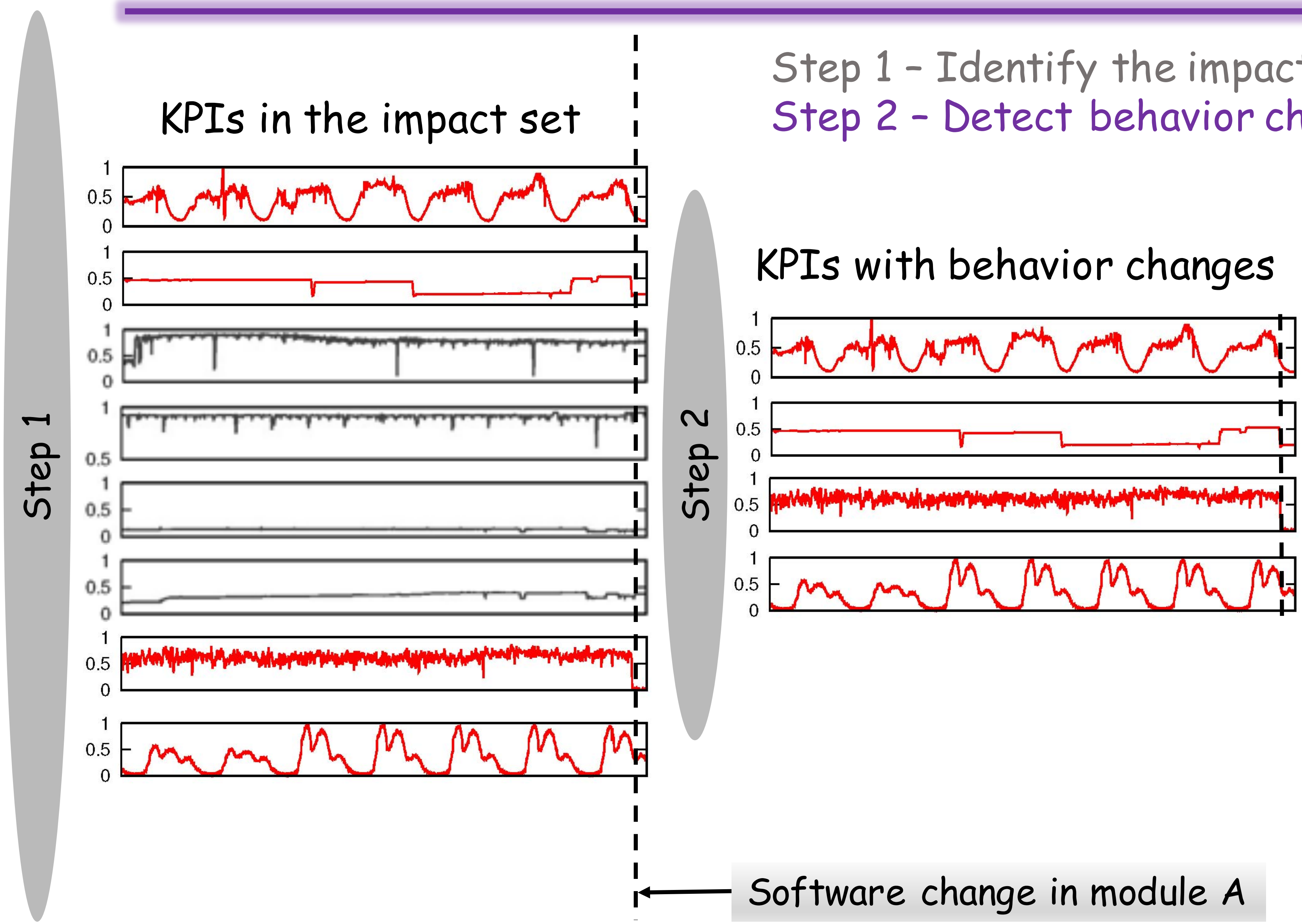


Software change in module A

Design Overview

Step 1 – Identify the impact set

Step 2 – Detect behavior changes in KPIs



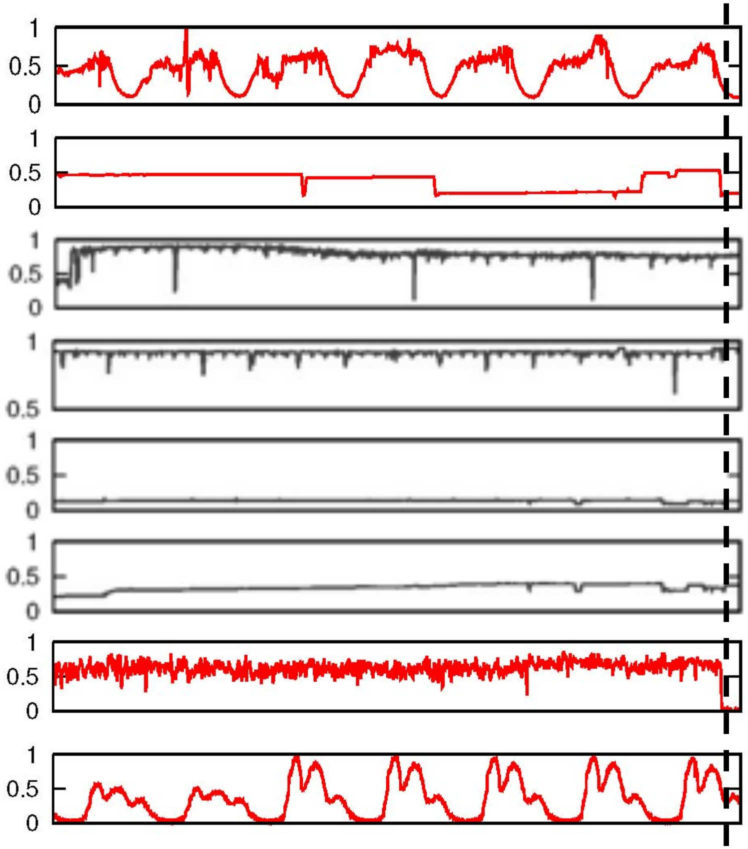
Design Overview

Step 1 – Identify the impact set

Step 2 – Detect behavior changes in KPIs

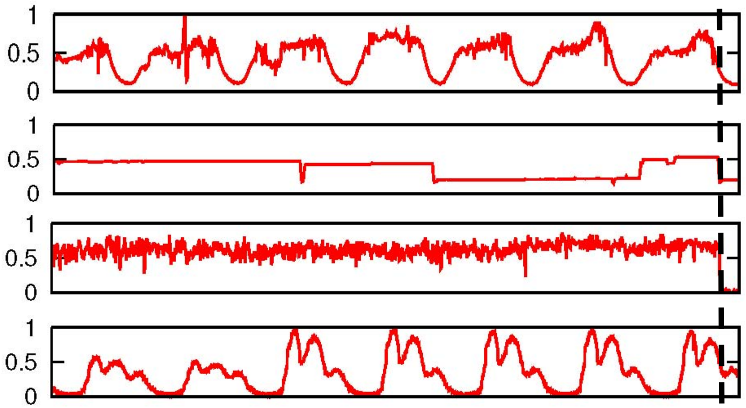
Step 1

KPIs in the impact set



Step 2

KPIs with behavior changes



Short detection delay requirement against robustness

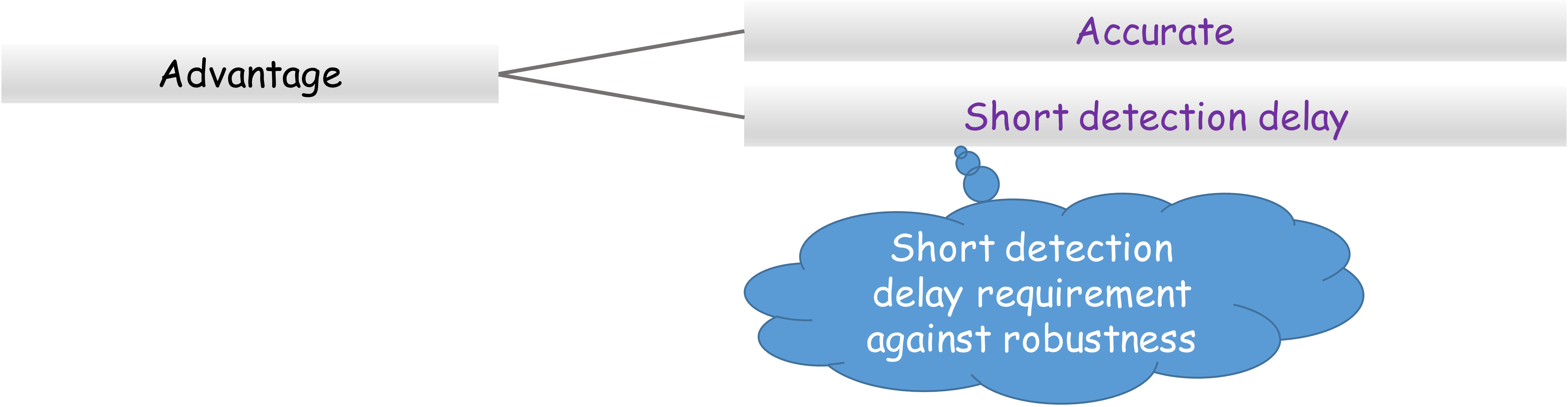
Diverse types of data

Large number of KPIs

Software change in module A

Improved Singular Spectrum Transform (SST)

- Improved singular spectrum transform (SST) $x_s(t) = 1 - \alpha(t)^T \beta(t)$



Advantage

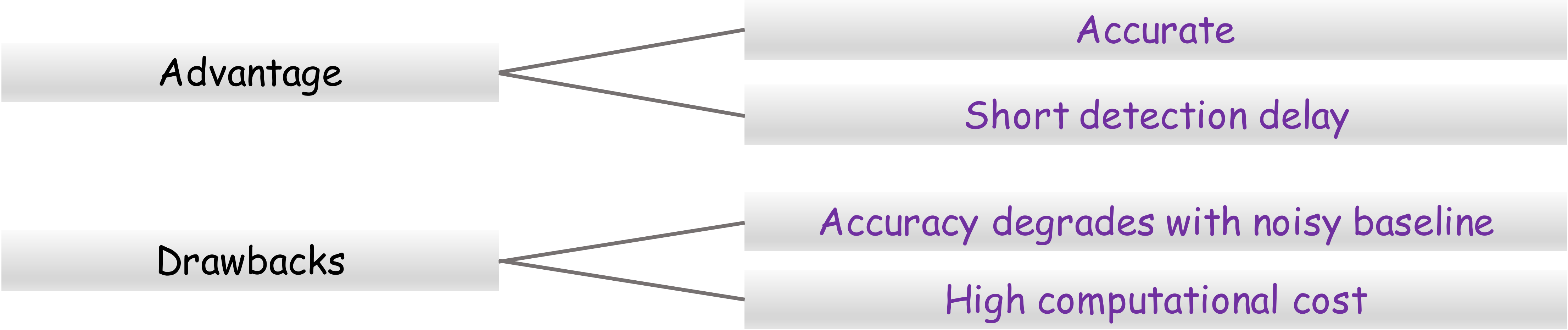
Accurate

Short detection delay

Short detection delay requirement against robustness

Improved Singular Spectrum Transform (SST)

- Improved singular spectrum transform (SST) $x_s(t) = 1 - \alpha(t)^T \beta(t)$



Advantage

Accurate

Short detection delay

Drawbacks

Accuracy degrades with noisy baseline

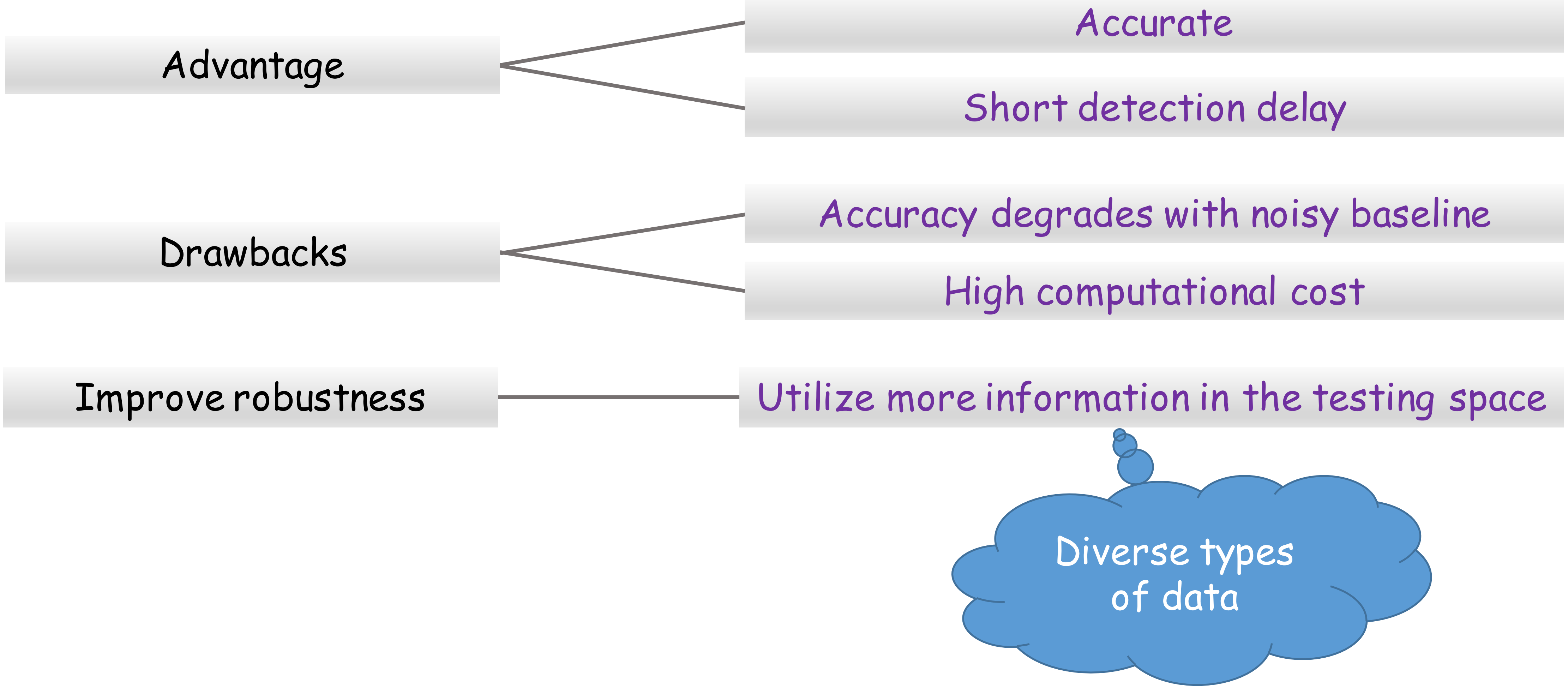
High computational cost

T. Idé and K. Tsuda, SDM 2007

Improved Singular Spectrum Transform (SST)

- Improved singular spectrum transform (SST)

$$\hat{x}(t) = \frac{\sum_{i=1}^{\eta} \lambda_i \times \varphi_i(t)}{\sum_{i=1}^{\eta} \lambda_i}$$



Advantage

Accurate

Short detection delay

Drawbacks

Accuracy degrades with noisy baseline

High computational cost

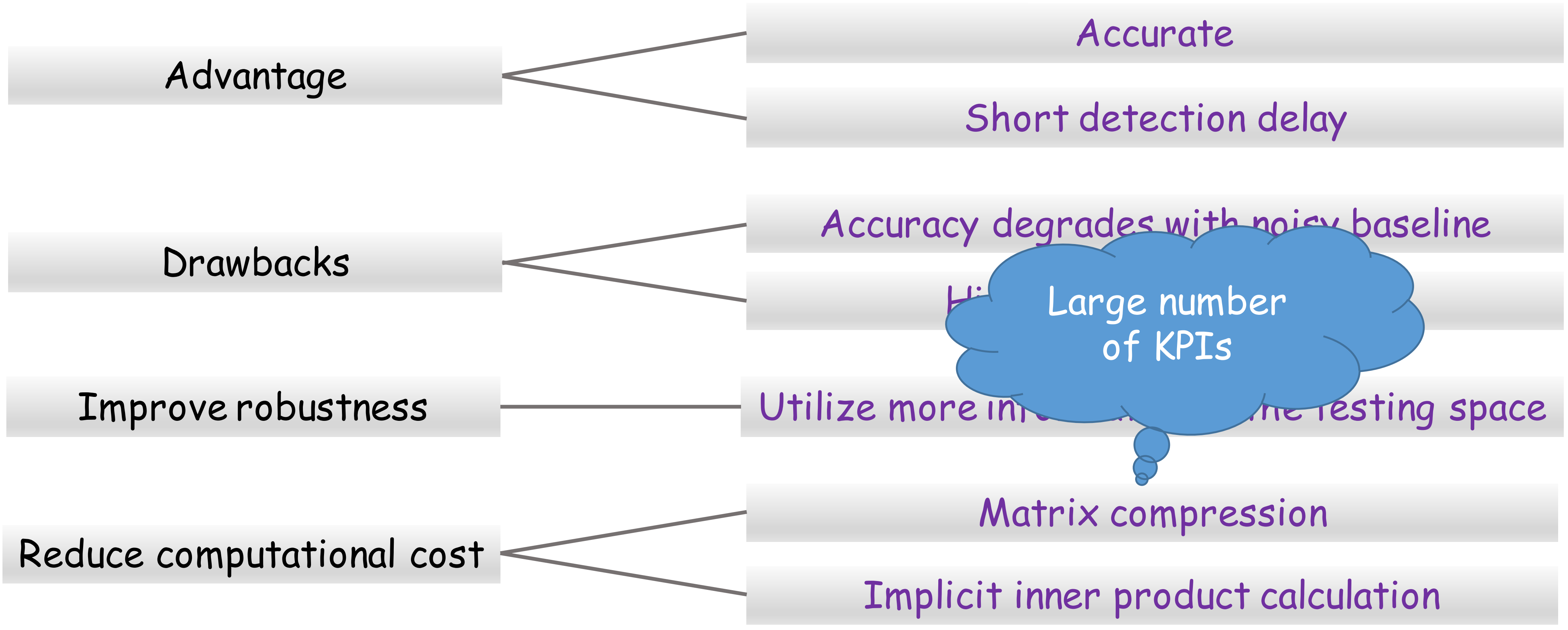
Improve robustness

Utilize more information in the testing space

Diverse types of data

Improved Singular Spectrum Transform (SST)

- Improved singular spectrum transform (SST) $\varphi_i(t) \simeq 1 - \sum_{j=1}^{\eta} {x_j}^2$



Advantage

- Accurate
- Short detection delay

Drawbacks

- Accuracy degrades with noisy baseline
- Hi

Improve robustness

- Utilize more inf ne testing space

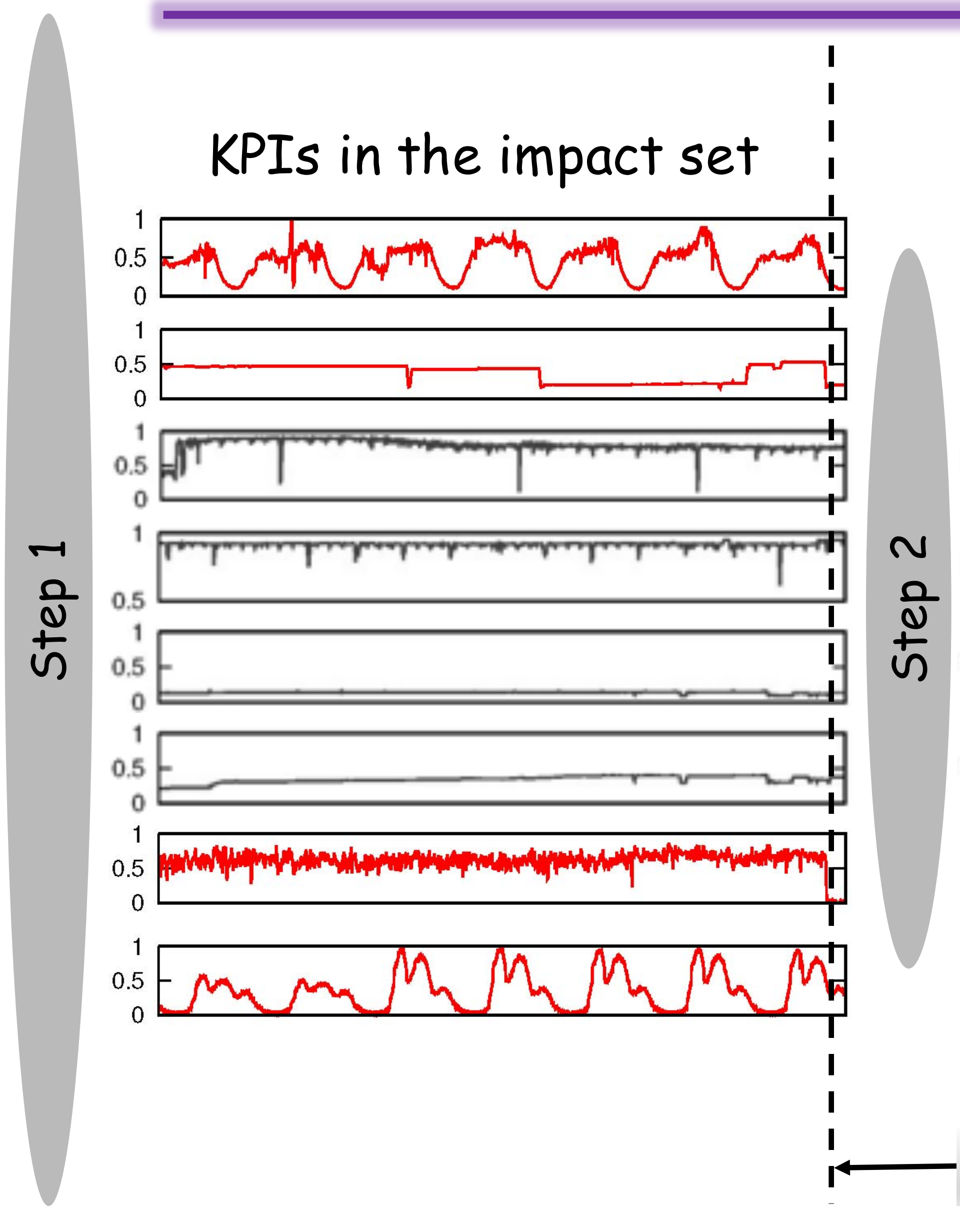
Large number of KPIs

Reduce computational cost

- Matrix compression
- Implicit inner product calculation

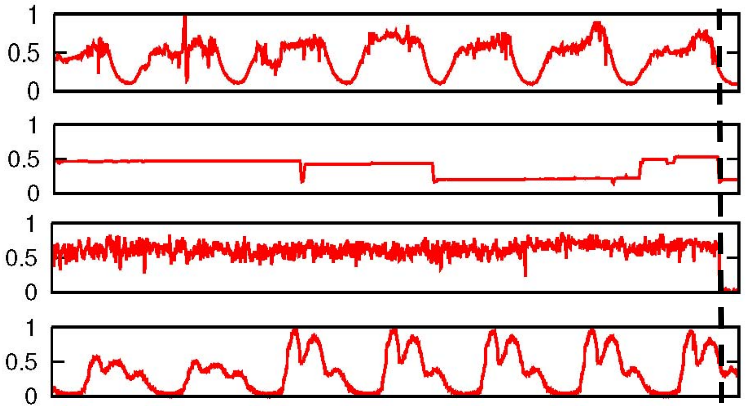
Design Overview

Step 1 – Identify the impact set
Step 2 – Detect behavior changes in KPIs
Step 3 – Eliminate KPI changes induced by other factors



KPIs in the impact set

KPIs with behavior changes



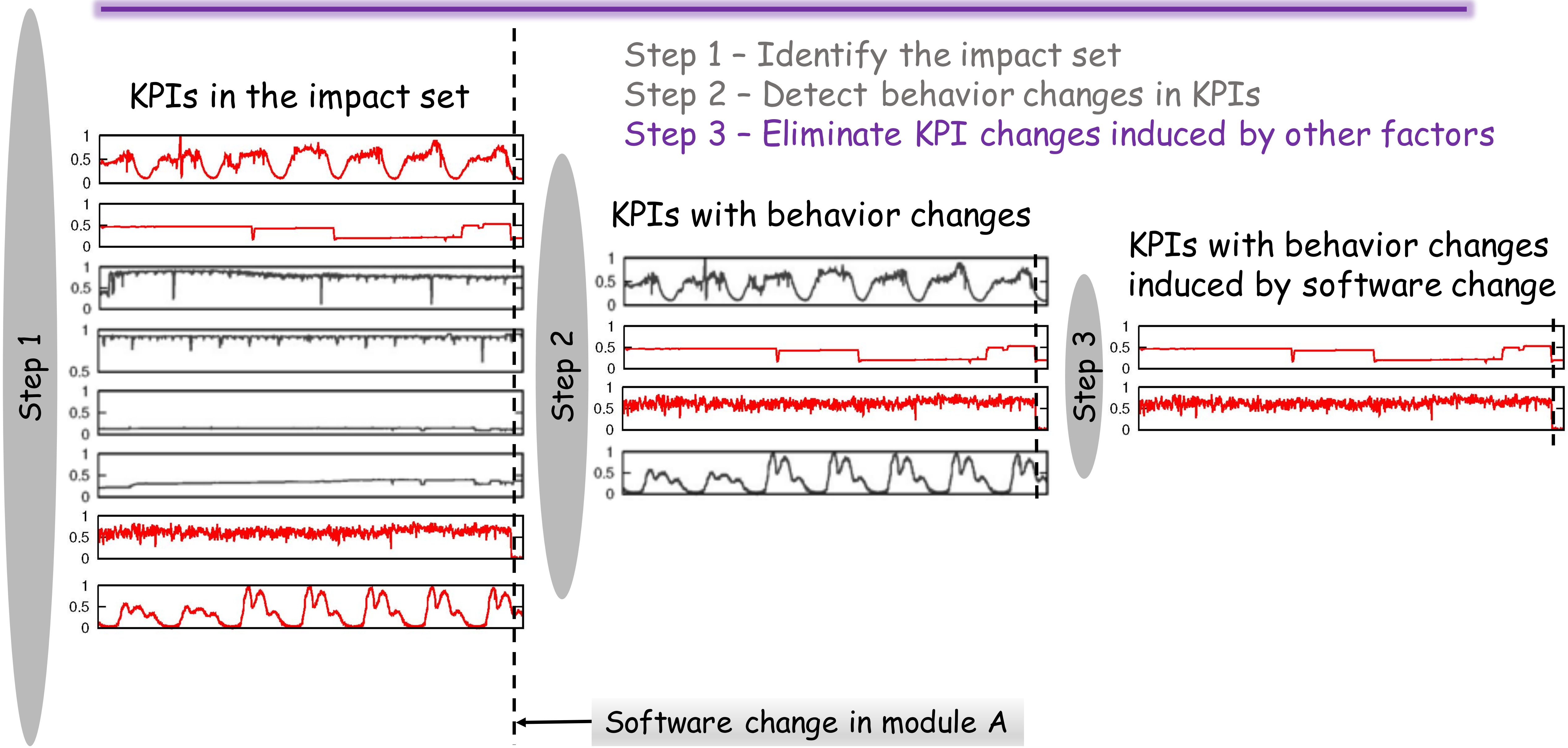
Software change in module A

Design Overview

Step 1 – Identify the impact set

Step 2 – Detect behavior changes in KPIs

Step 3 – Eliminate KPI changes induced by other factors



KPIs in the impact set

KPIs with behavior changes

KPIs with behavior changes induced by software change

Software change in module A

Design Overview

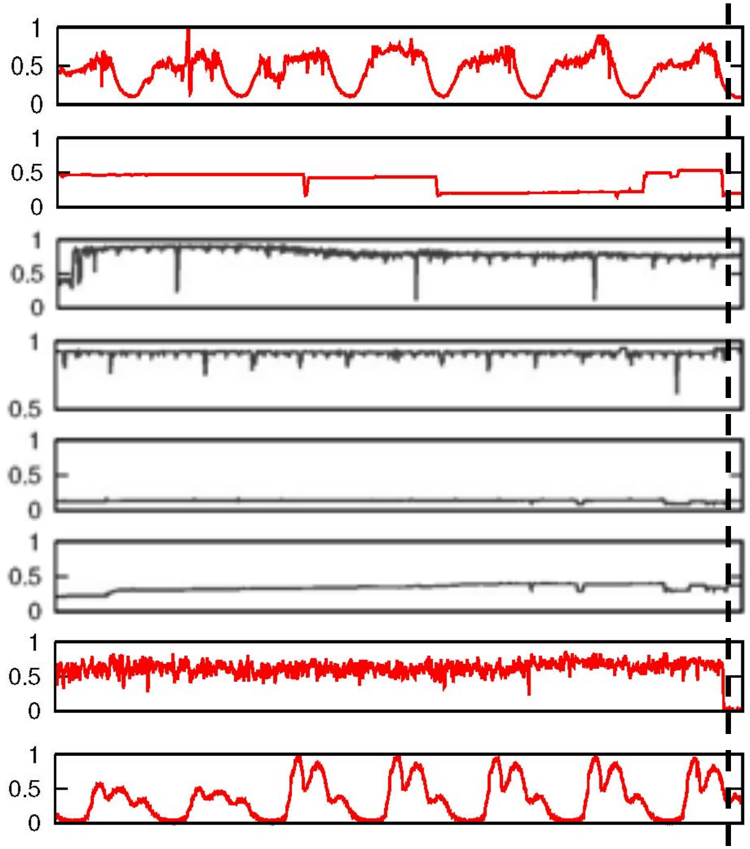
Step 1 – Identify the impact set

Step 2 – Detect behavior changes in KPIs

Step 3 – Eliminate KPI changes induced by other factors

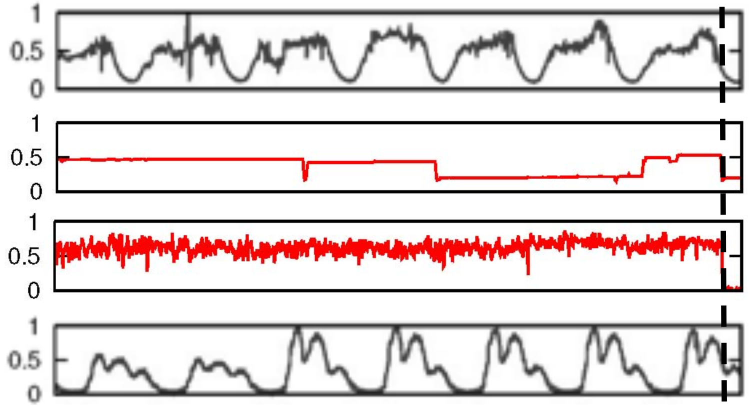
Step 1

KPIs in the impact set



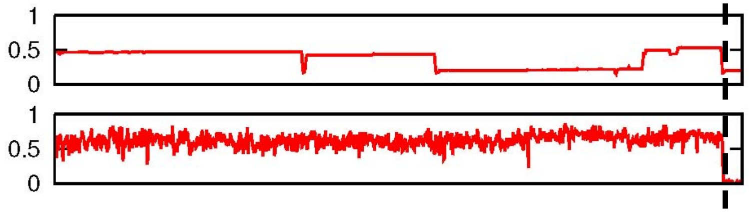
Step 2

KPIs with behavior changes



Step 3

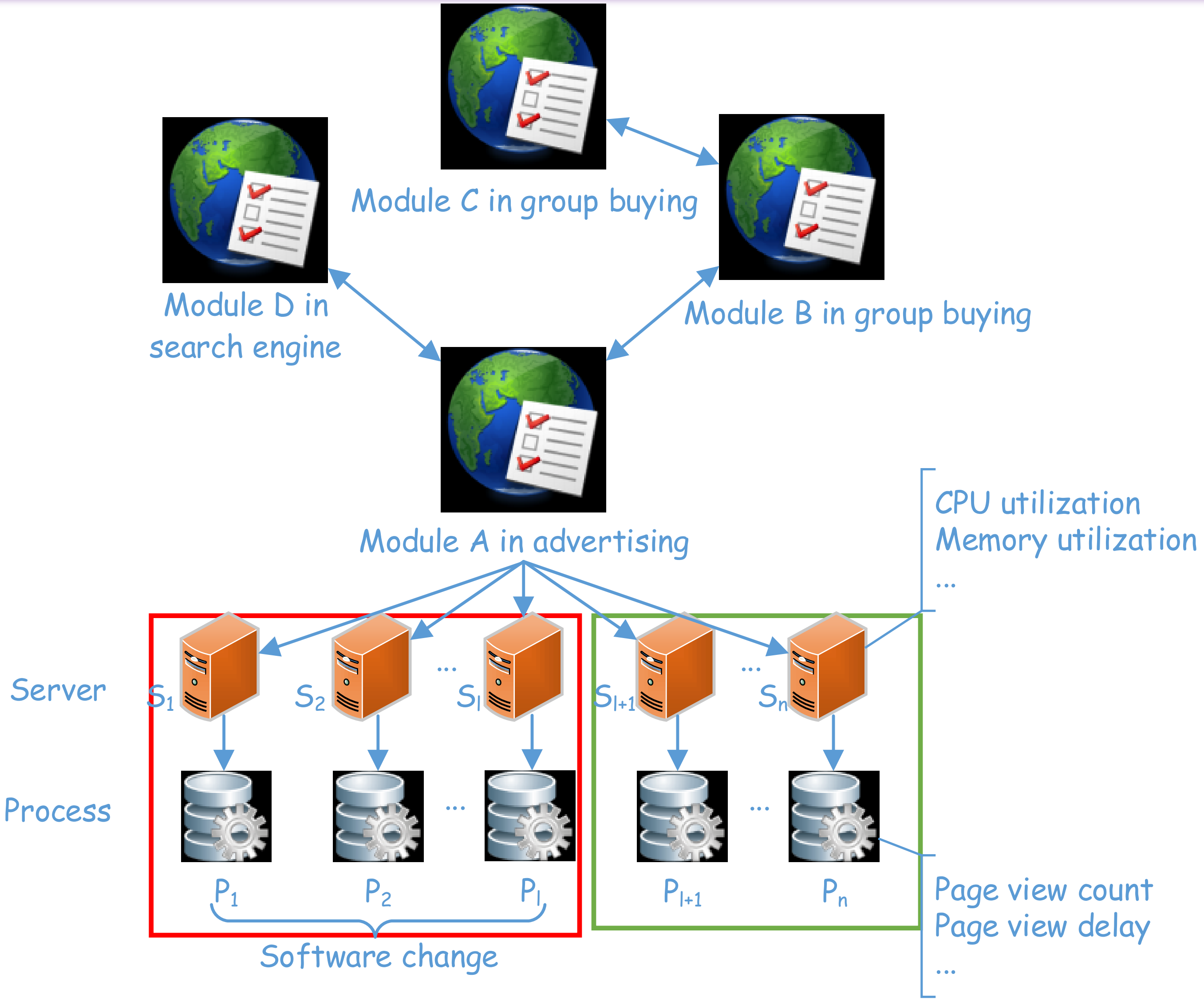
KPIs with behavior changes induced by software change



KPI changes maybe caused by other factors

Software change in module A

Eliminate KPI Changes Induced by Other Factors

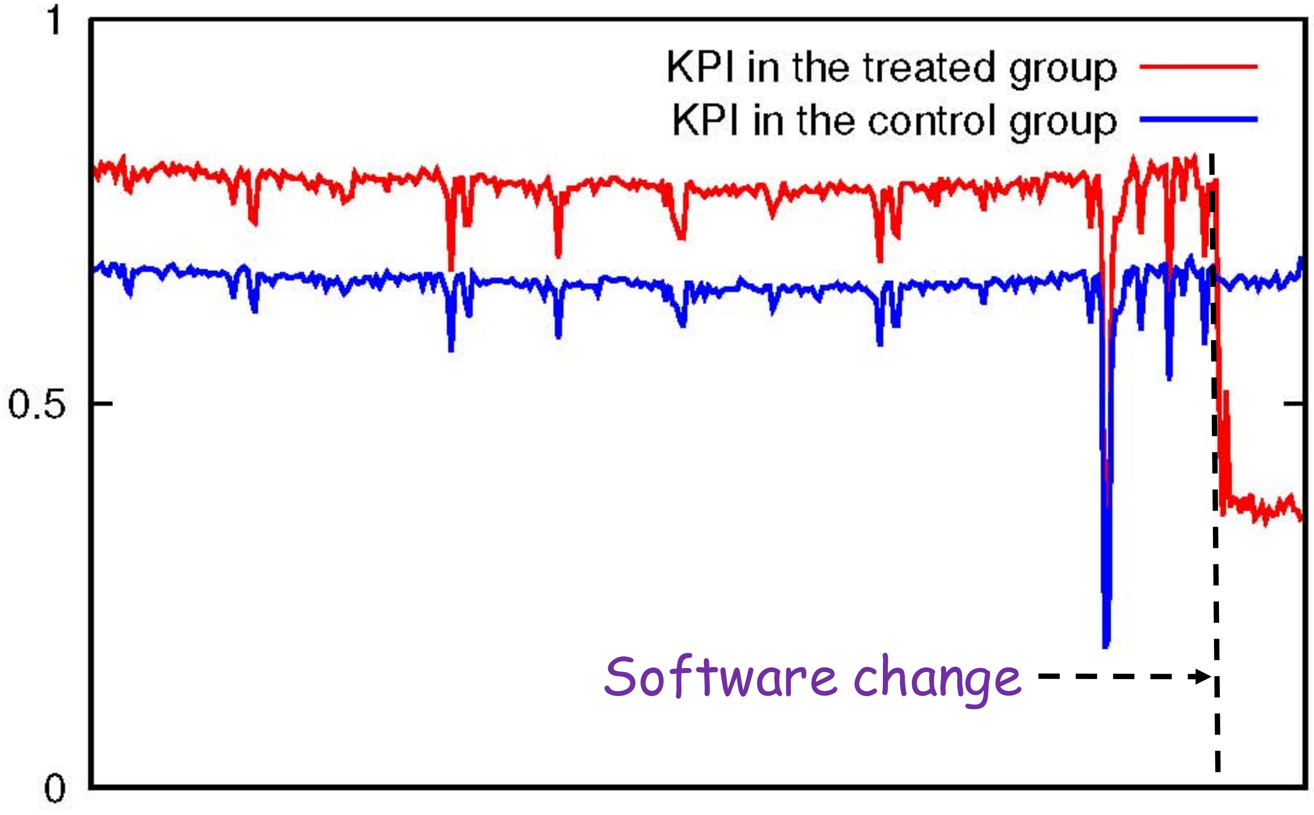


Eliminate KPI Changes Induced by Other Factors

- Split testing
    - Evaluation of interventions instituted at a specific time
    - Control group & treated group

Eliminate KPI Changes Induced by Other Factors

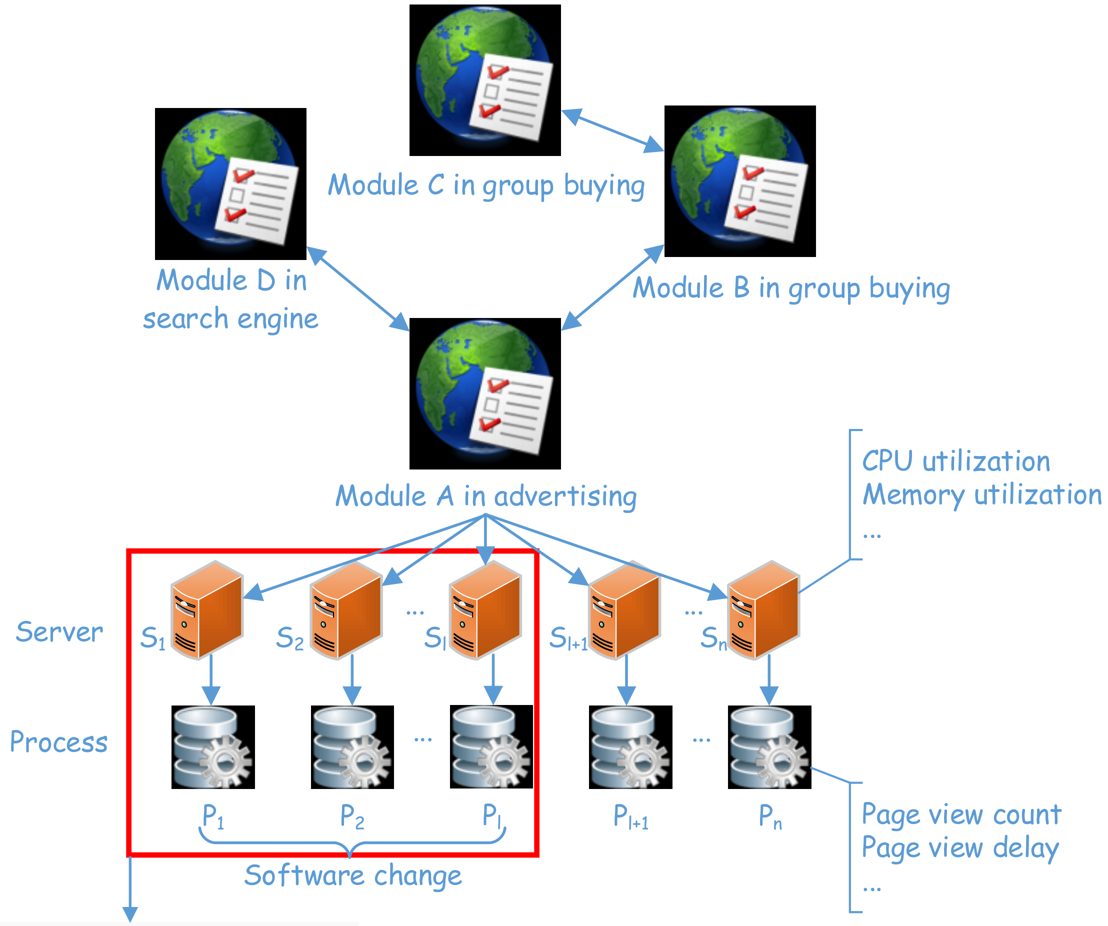
- Split testing
  - Evaluation of interventions instituted at a specific time
  - Control group & treated group



Eliminate KPI Changes Induced by Other Factors

Treated group

- Servers/processes in the impact set



treated group

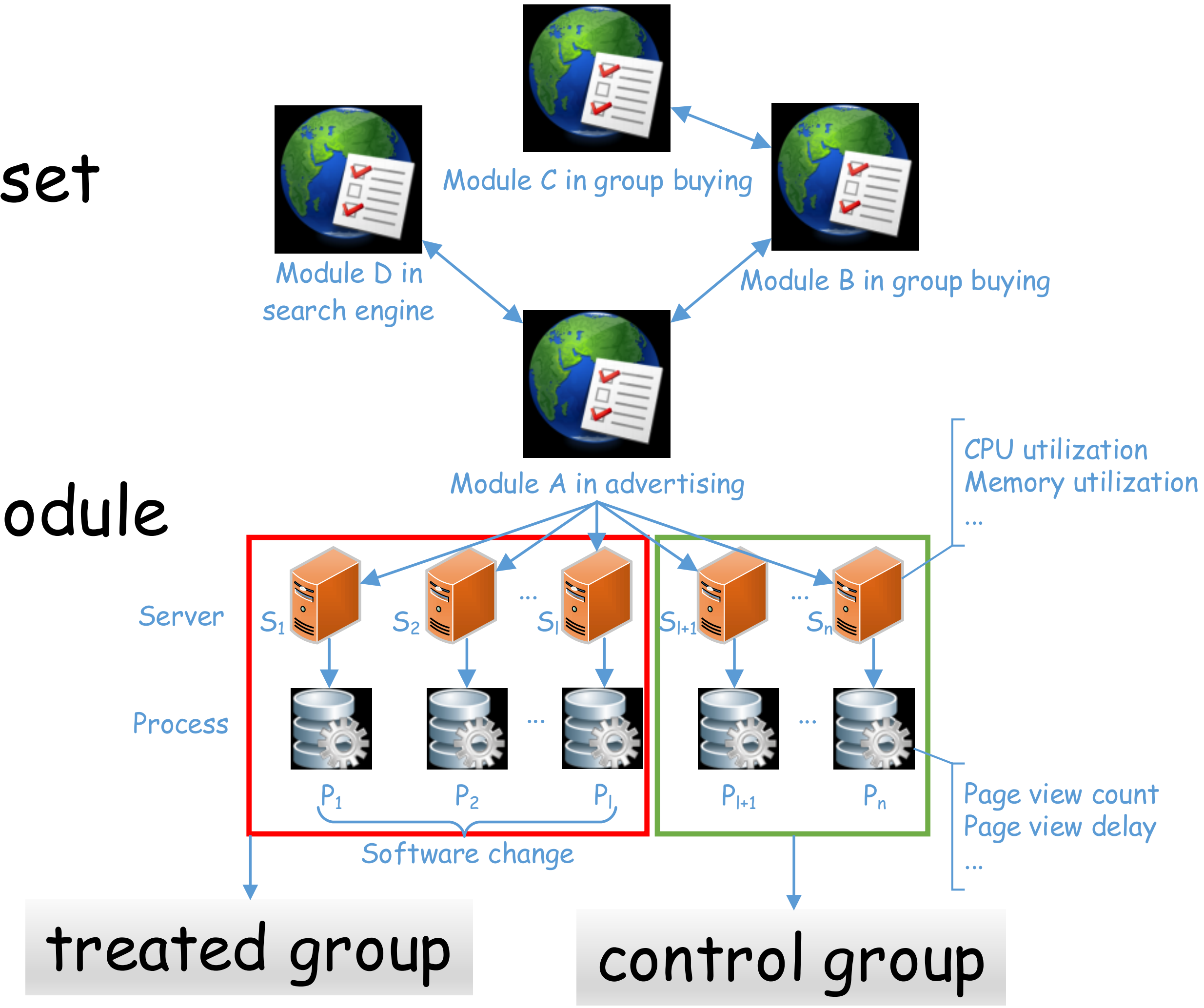
Eliminate KPI Changes Induced by Other Factors

Treated group

- Servers/processes in the impact set

Control group

- Servers/processes in the same module
- Without software change



Eliminate KPI Changes Induced by Other Factors

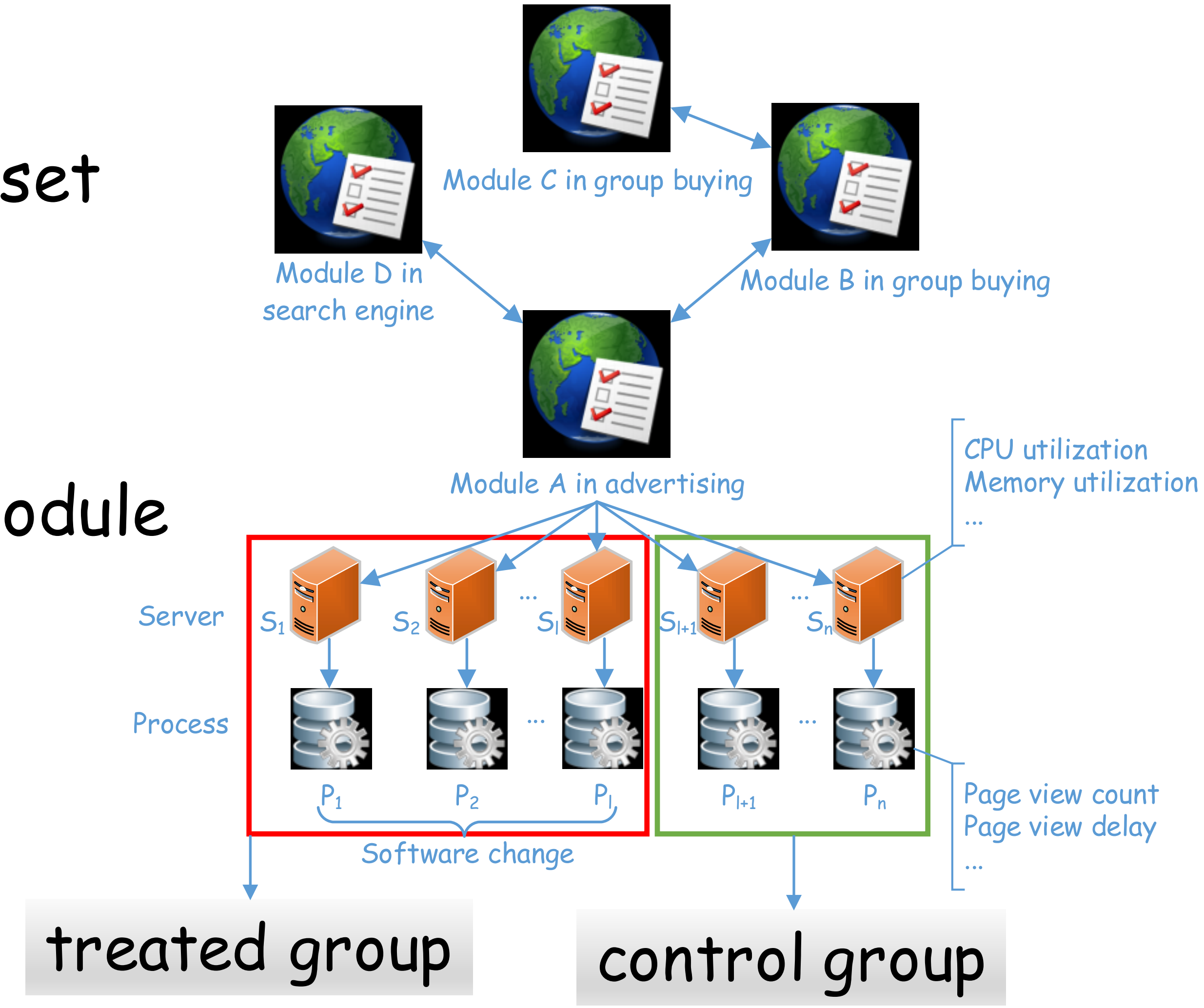
Treated group

- Servers/processes in the impact set

Control group

- Servers/processes in the same module
- Without software change

DiD method



Eliminate KPI Changes Induced by Other Factors

Treated group

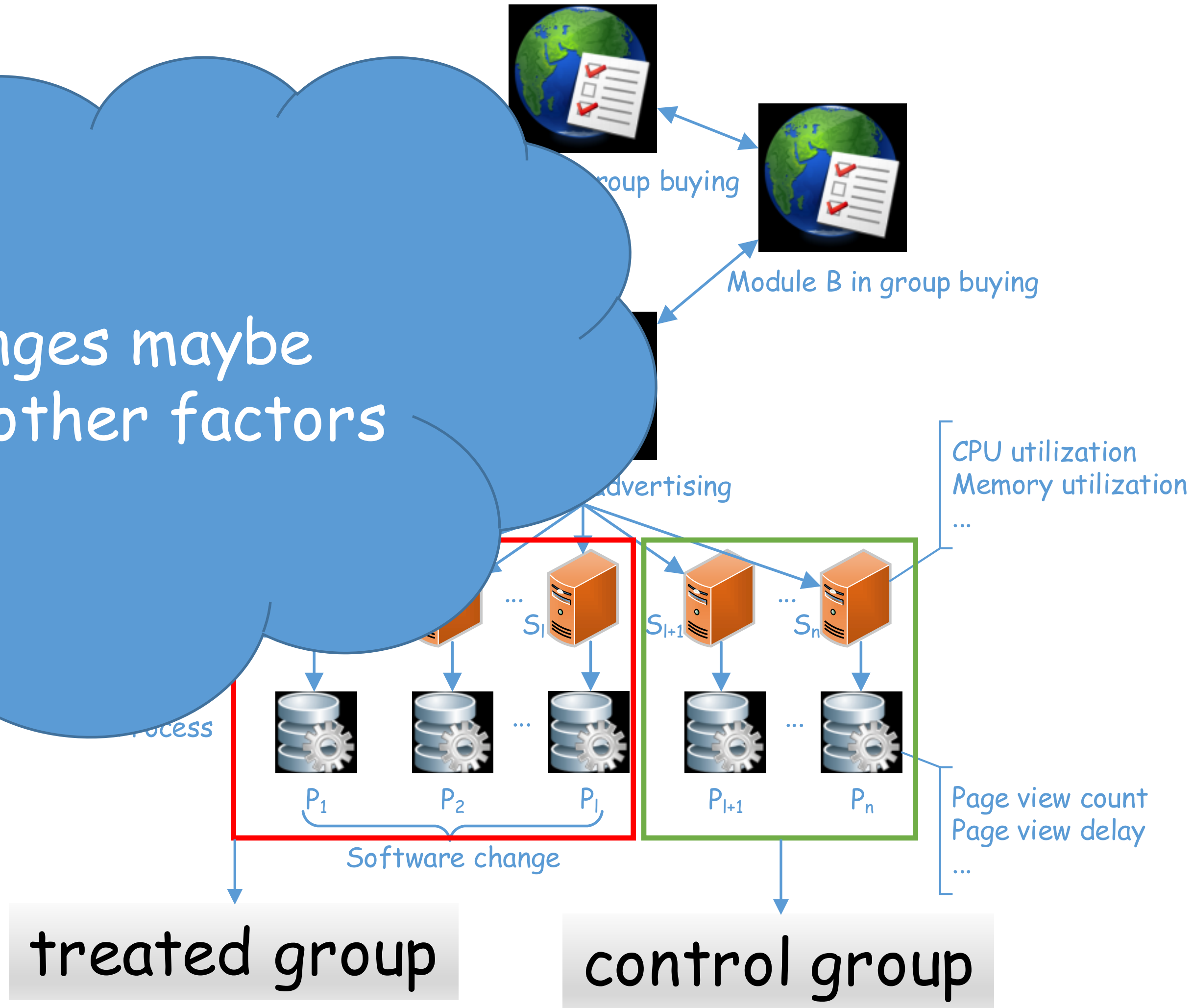
- Servers/processes in t

Control group

- Servers/processes
- Without software cha

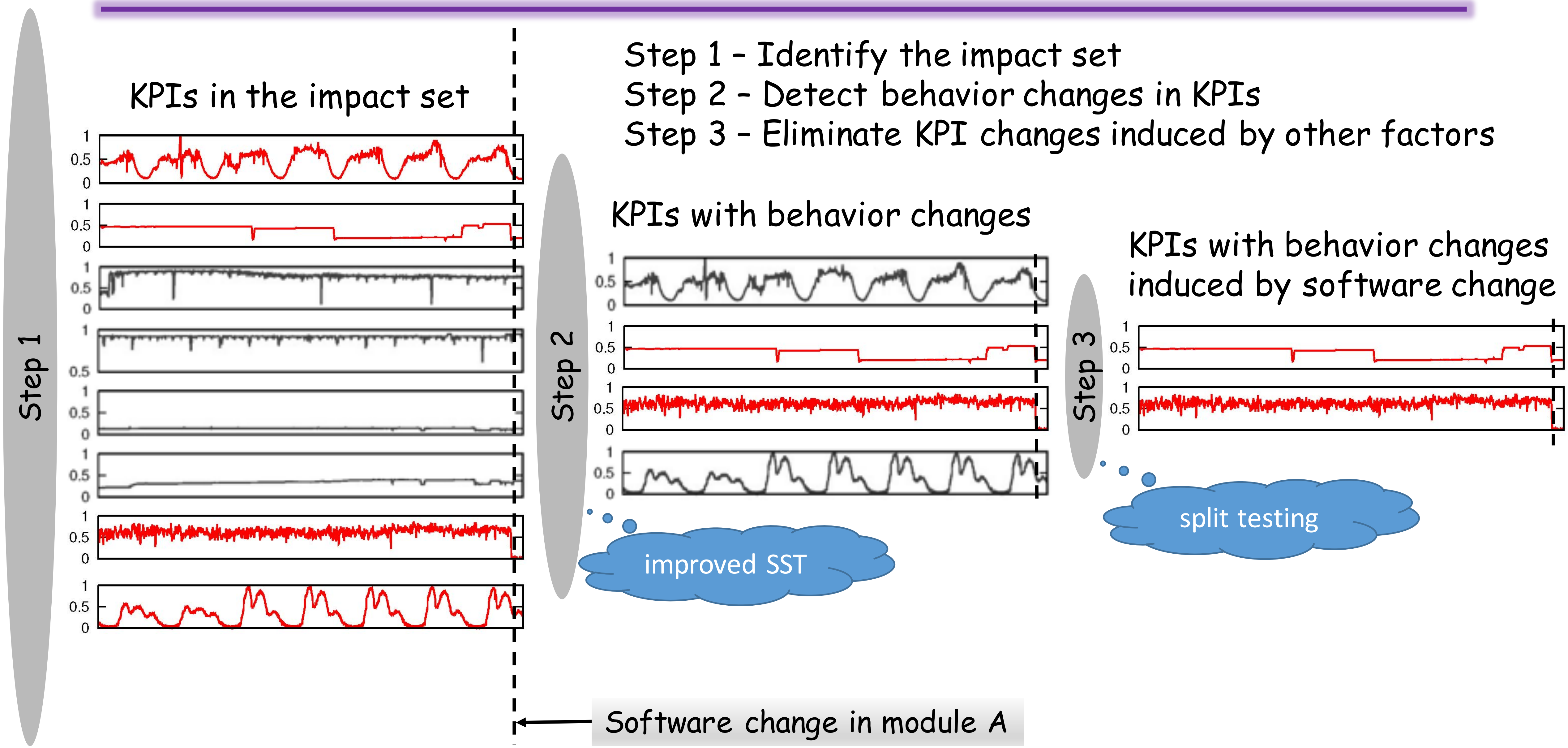
DiD method

KPI changes maybe caused by other factors



Design Overview

Step 1 – Identify the impact set
Step 2 – Detect behavior changes in KPIs
Step 3 – Eliminate KPI changes induced by other factors



Step 1

KPIs in the impact set

Step 2

KPIs with behavior changes

improved SST

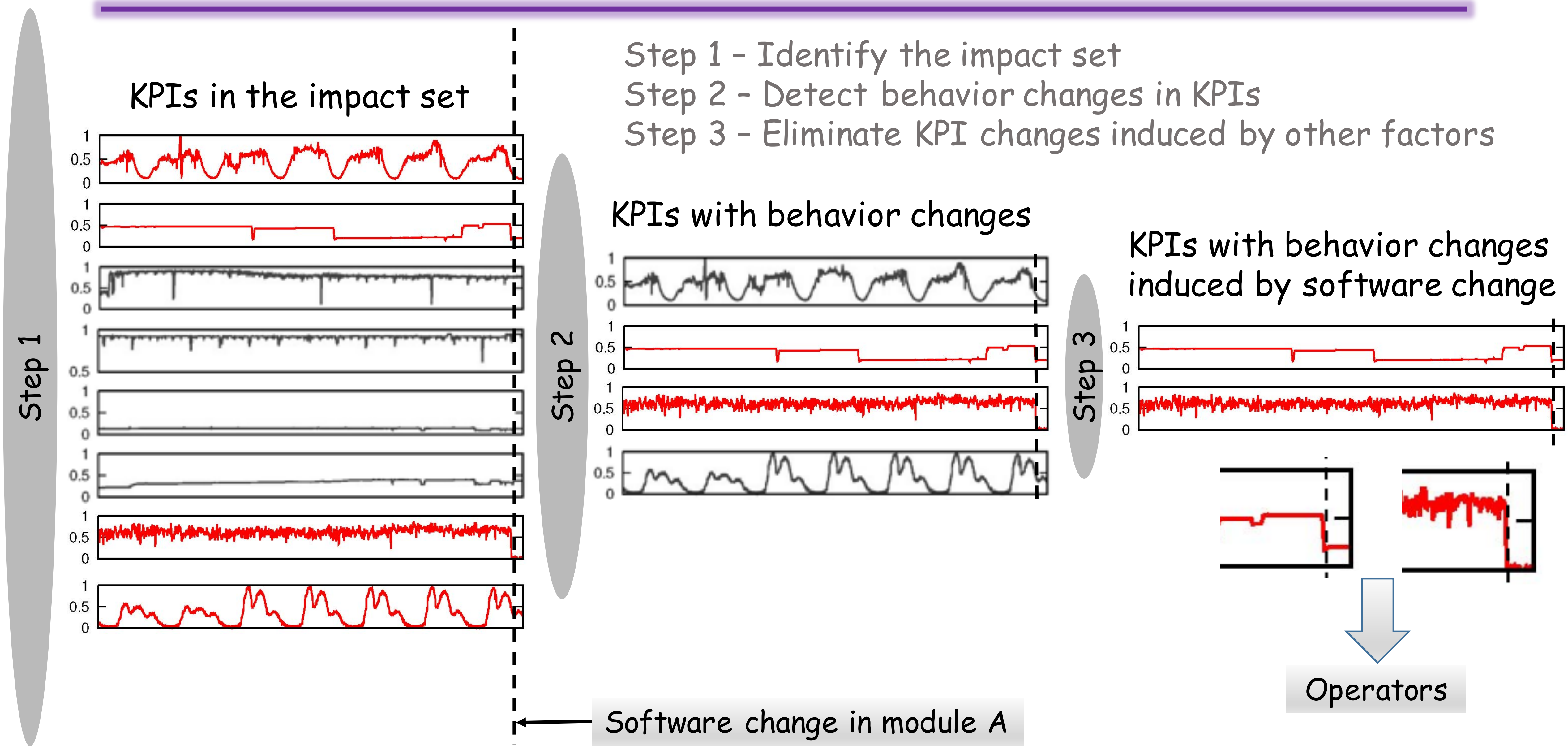
Step 3

KPIs with behavior changes induced by software change

split testing

Software change in module A

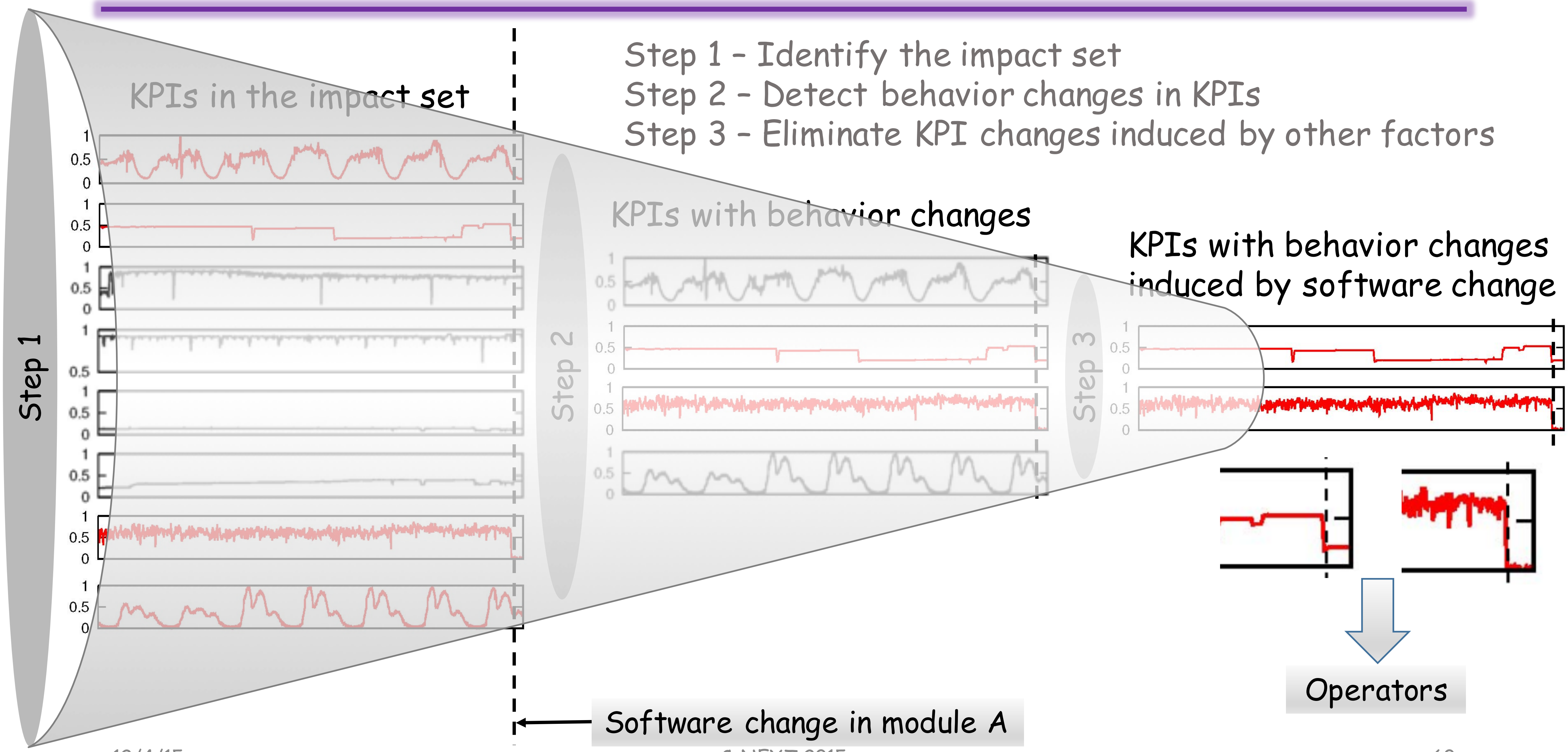
Design Overview



Step 1 – Identify the impact set
Step 2 – Detect behavior changes in KPIs
Step 3 – Eliminate KPI changes induced by other factors

Step 1
KPIs in the impact set
Step 2
KPIs with behavior changes
Step 3
KPIs with behavior changes induced by software change
1
0.5
0
Operators
Software change in module A

Design Overview

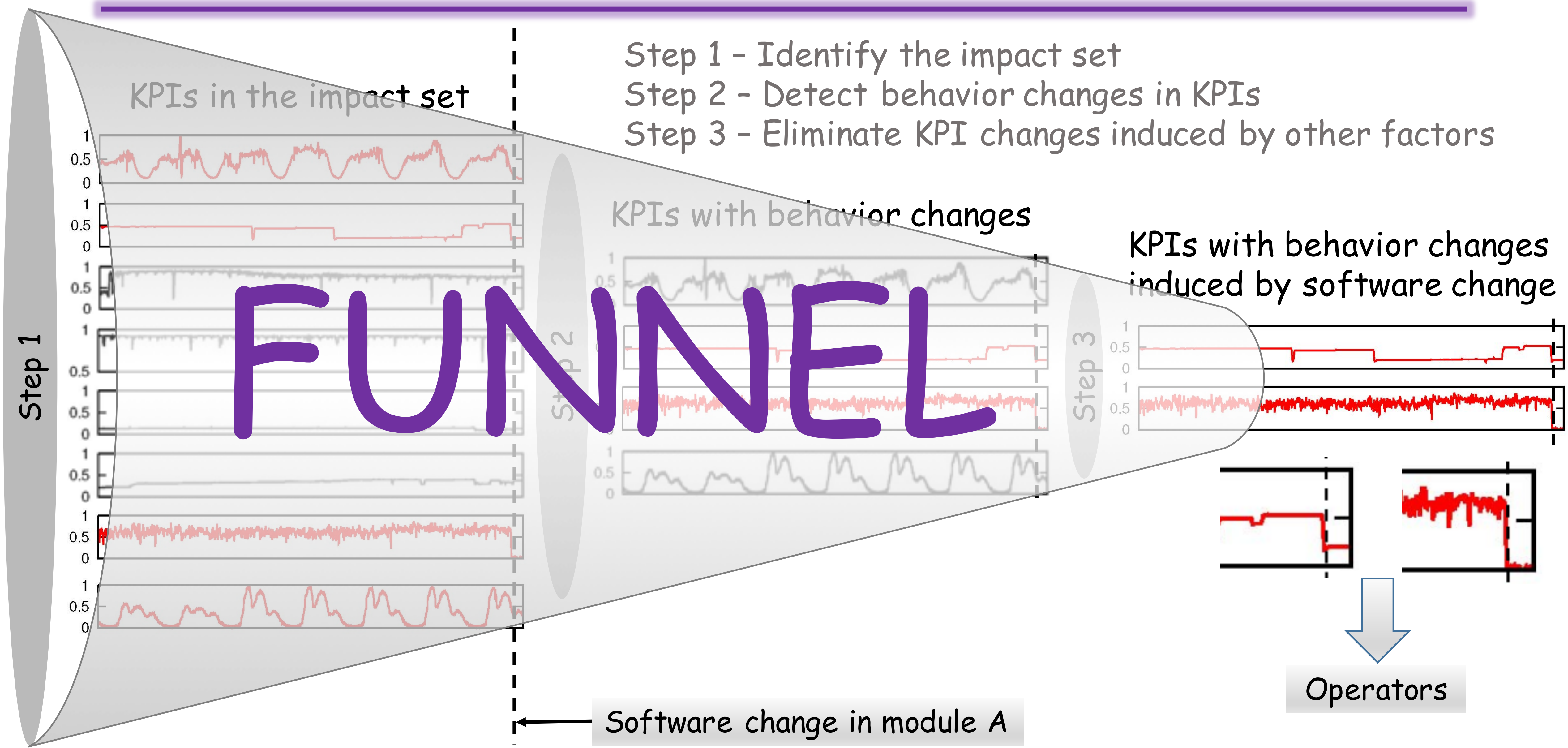


Step 1 – Identify the impact set
Step 2 – Detect behavior changes in KPIs
Step 3 – Eliminate KPI changes induced by other factors

Step 1
KPIs in the impact set
1
0.5
0
Step 2
KPIs with behavior changes
Step 3
KPIs with behavior changes induced by software change
Operators
Software change in module A

Design Overview

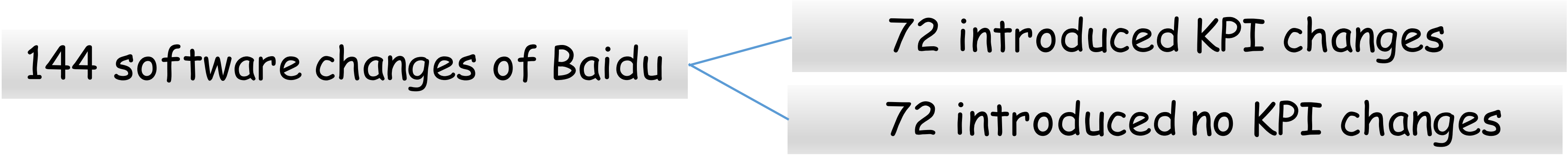
Step 1 – Identify the impact set
Step 2 – Detect behavior changes in KPIs
Step 3 – Eliminate KPI changes induced by other factors


KPIs in the impact set
Step 1
Step 2
Step 3
FUNNEL
KPIs with behavior changes
KPIs with behavior changes induced by software change
1
0.5
0
Operators
Software change in module A

Outline

- Background and Motivation
- Challenges
- Key Ideas
- Results
- Conclusion

Datasets of Evaluation

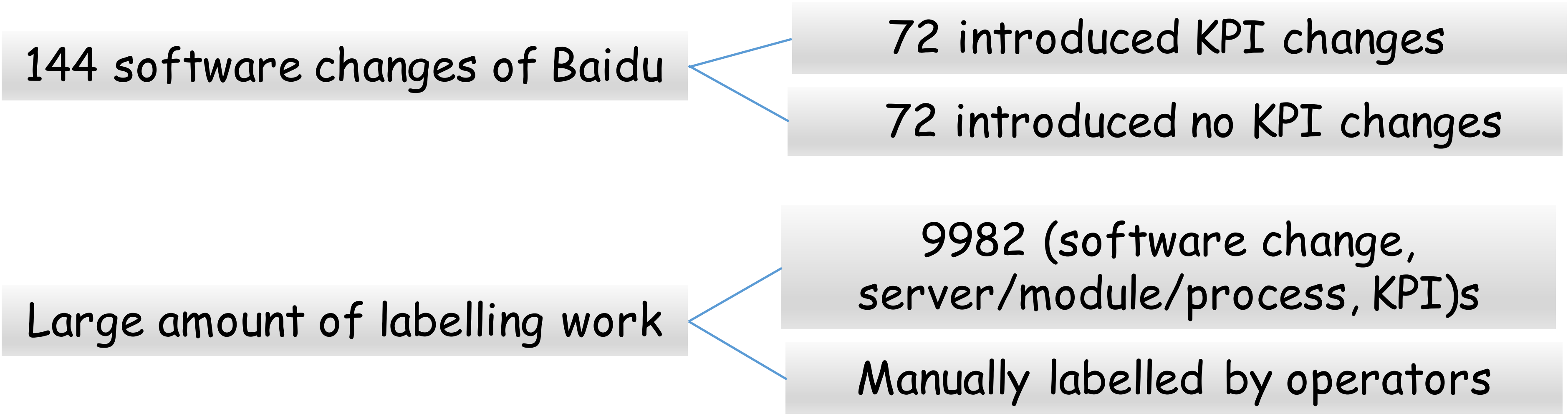


144 software changes of Baidu

72 introduced KPI changes

72 introduced no KPI changes

Datasets of Evaluation



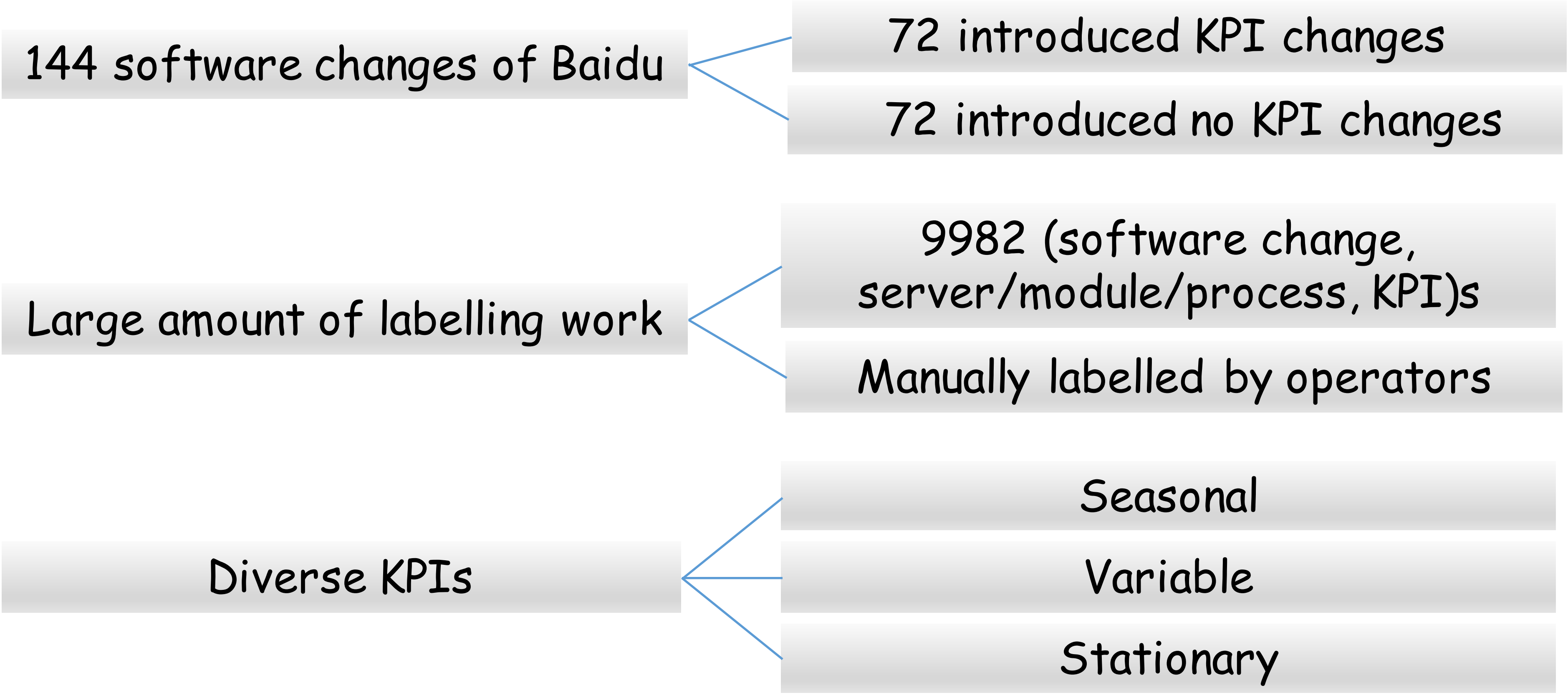
144 software changes of Baidu

- 72 introduced KPI changes
- 72 introduced no KPI changes

Large amount of labelling work

- 9982 (software change, server/module/process, KPI)s
- Manually labelled by operators

Datasets of Evaluation



144 software changes of Baidu

- 72 introduced KPI changes
- 72 introduced no KPI changes

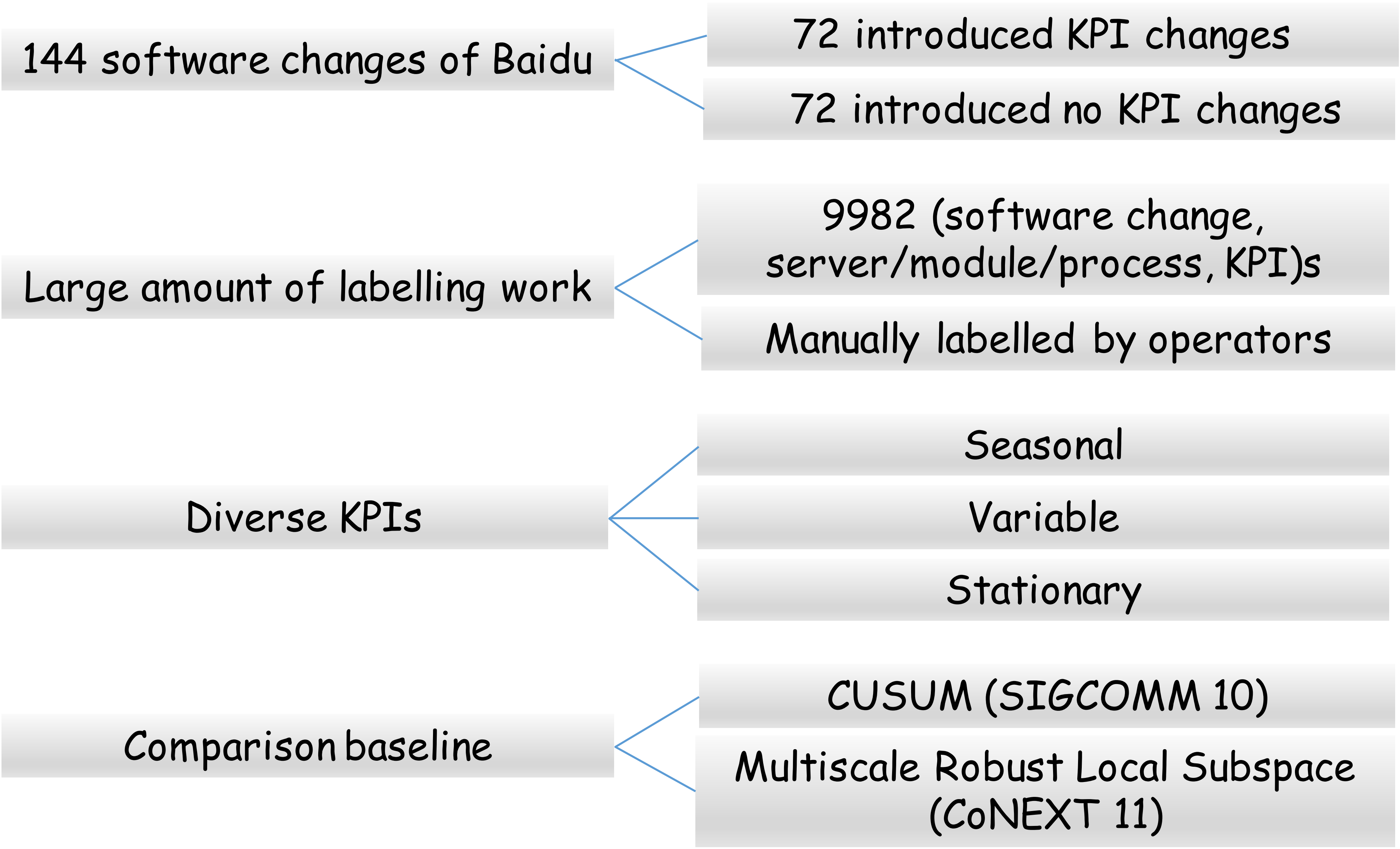
Large amount of labelling work

- 9982 (software change, server/module/process, KPI)s
- Manually labelled by operators

Diverse KPIs

- Seasonal
- Variable
- Stationary

Datasets of Evaluation



144 software changes of Baidu

- 72 introduced KPI changes
- 72 introduced no KPI changes

Large amount of labelling work

- 9982 (software change, server/module/process, KPI)s
- Manually labelled by operators

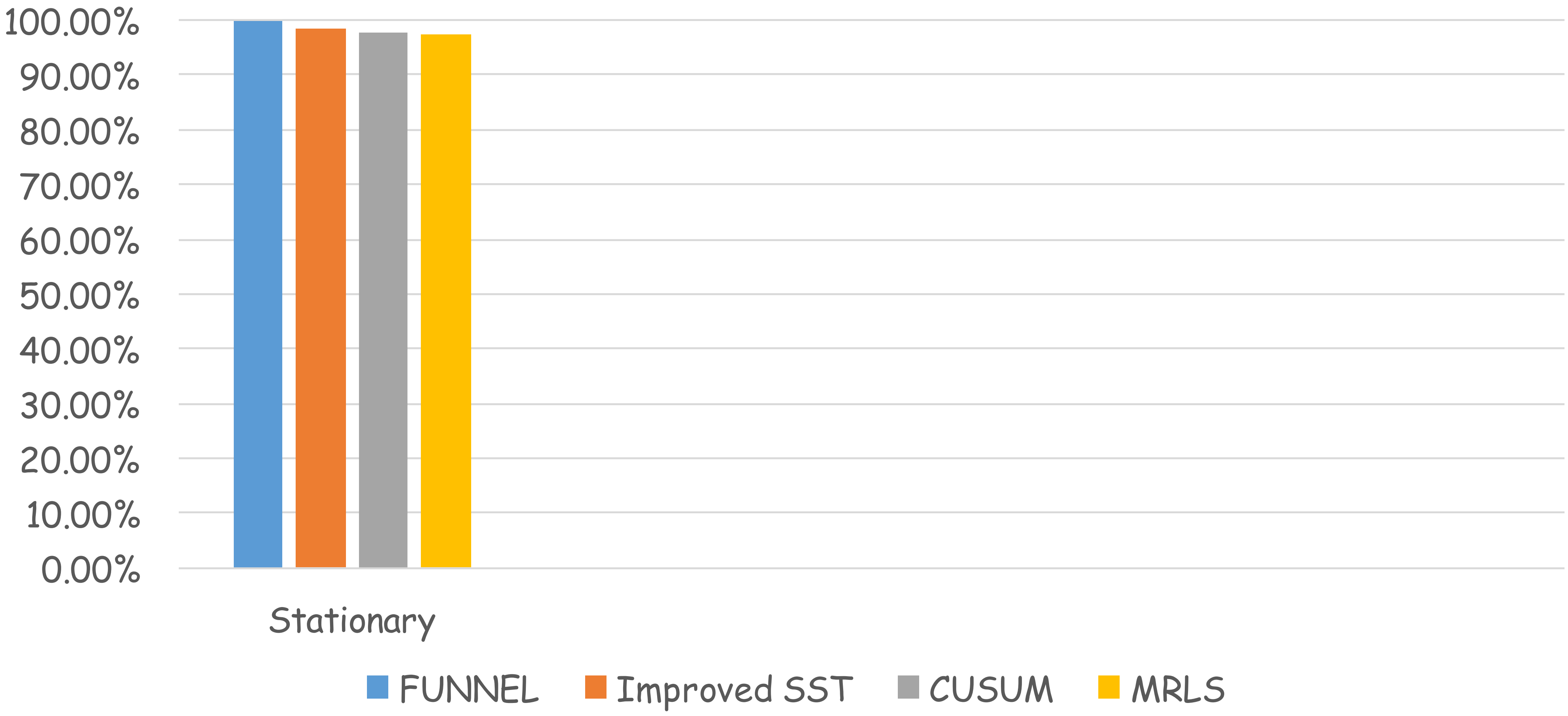
Diverse KPIs

- Seasonal
- Variable
- Stationary

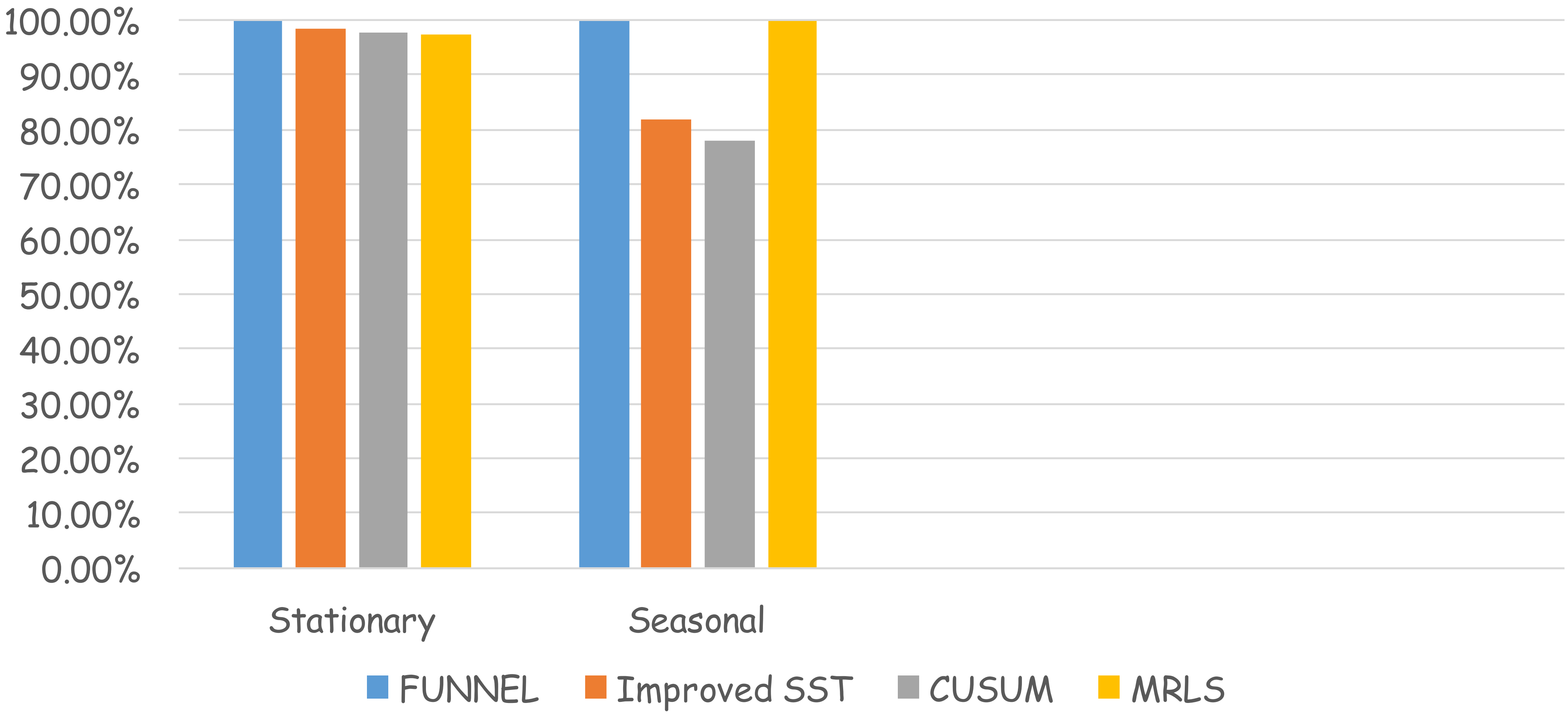
Comparison baseline

- CUSUM (SIGCOMM 10)
- Multiscale Robust Local Subspace (CoNEXT 11)

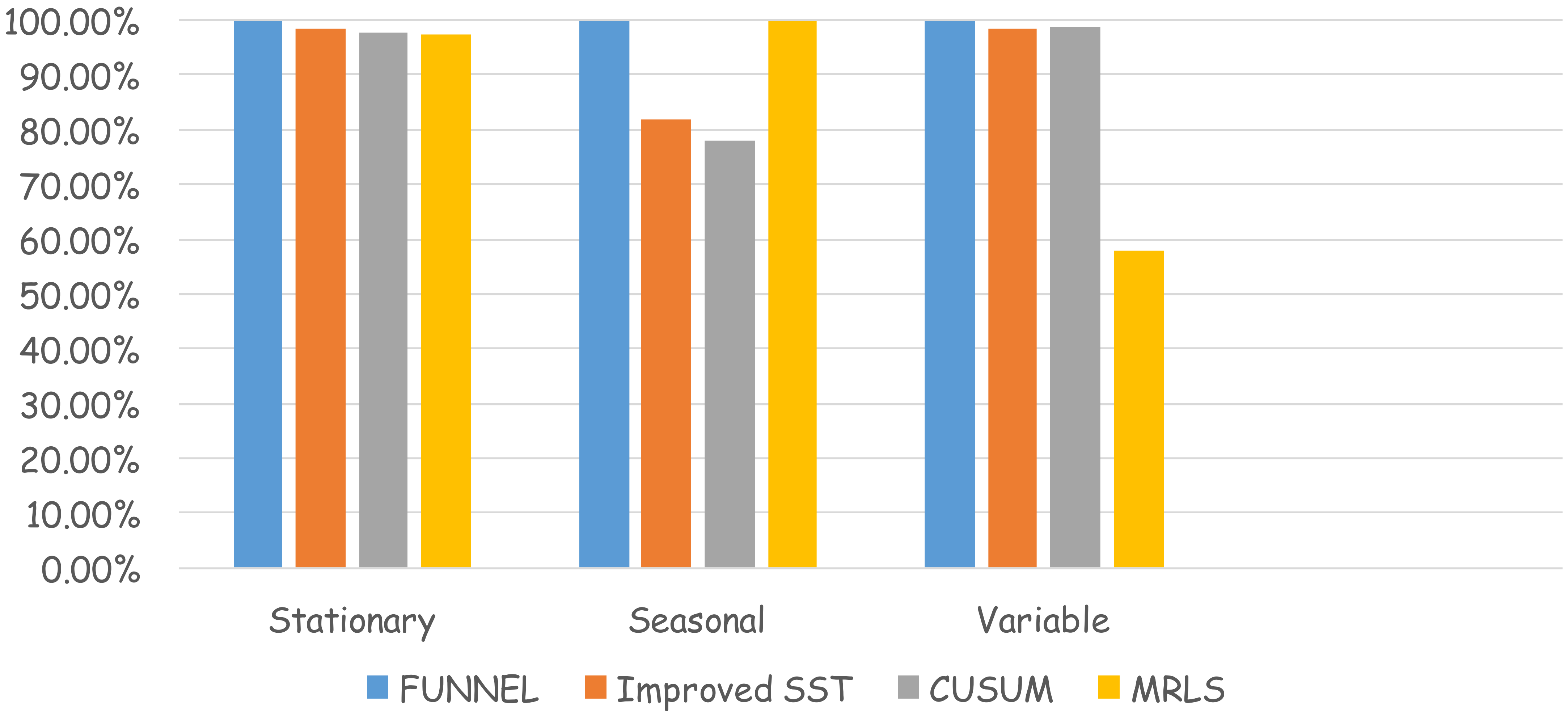
Comparison of Accuracy



Comparison of Accuracy



Comparison of Accuracy



Comparison of Computational Cost

- Real-world scenario
  - At least 1 million KPIs need to be monitored
  - The detection interval for each KPI is 1 minute
  - Runs on the same kinds of CPU as testing

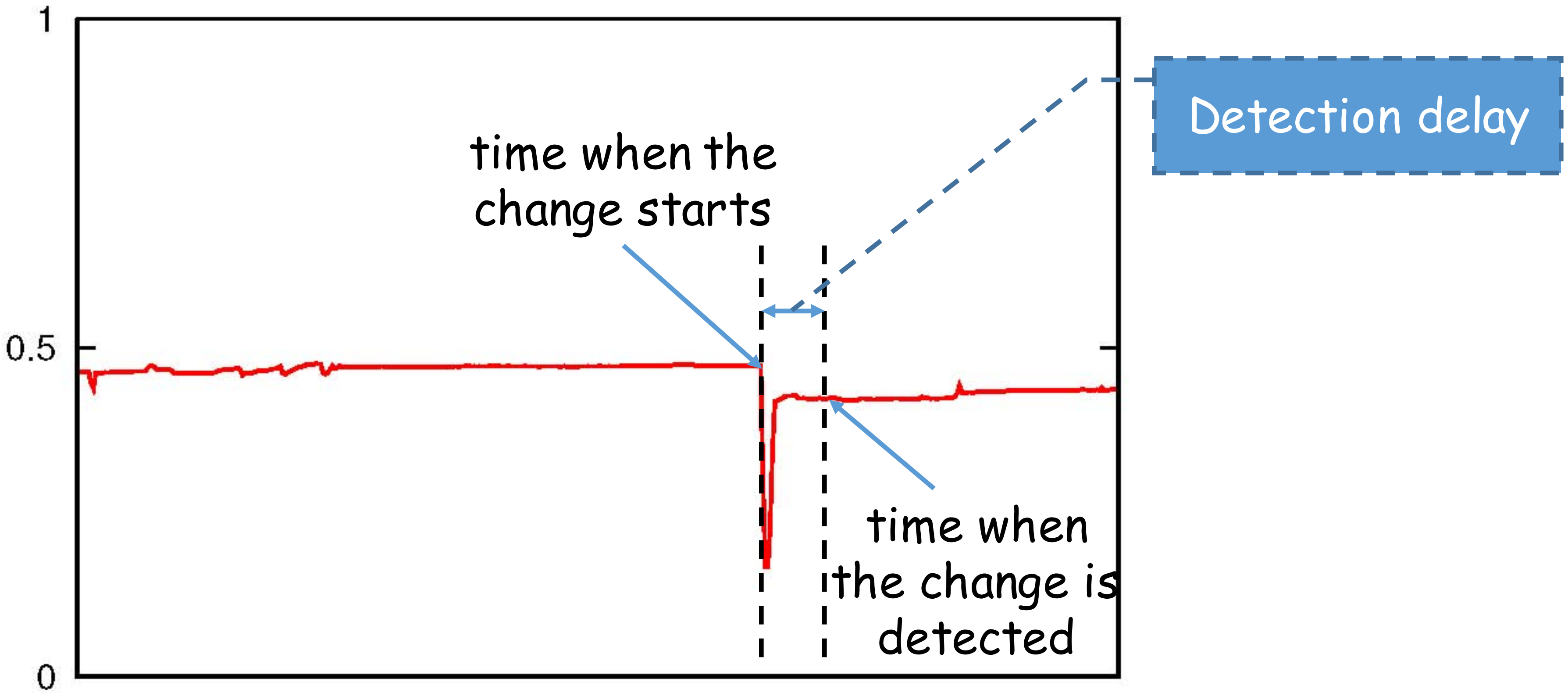
Comparison of Computational Cost

- Real-world scenario
  - At least 1 million KPIs need to be monitored
  - Each KPI is detected every 1 minute
  - Runs on the same kinds of CPU as testing
- Comparison results

Method	FUNNEL	CUSUM	MRLS
Number of cores for one million KPIs	7	31	47526

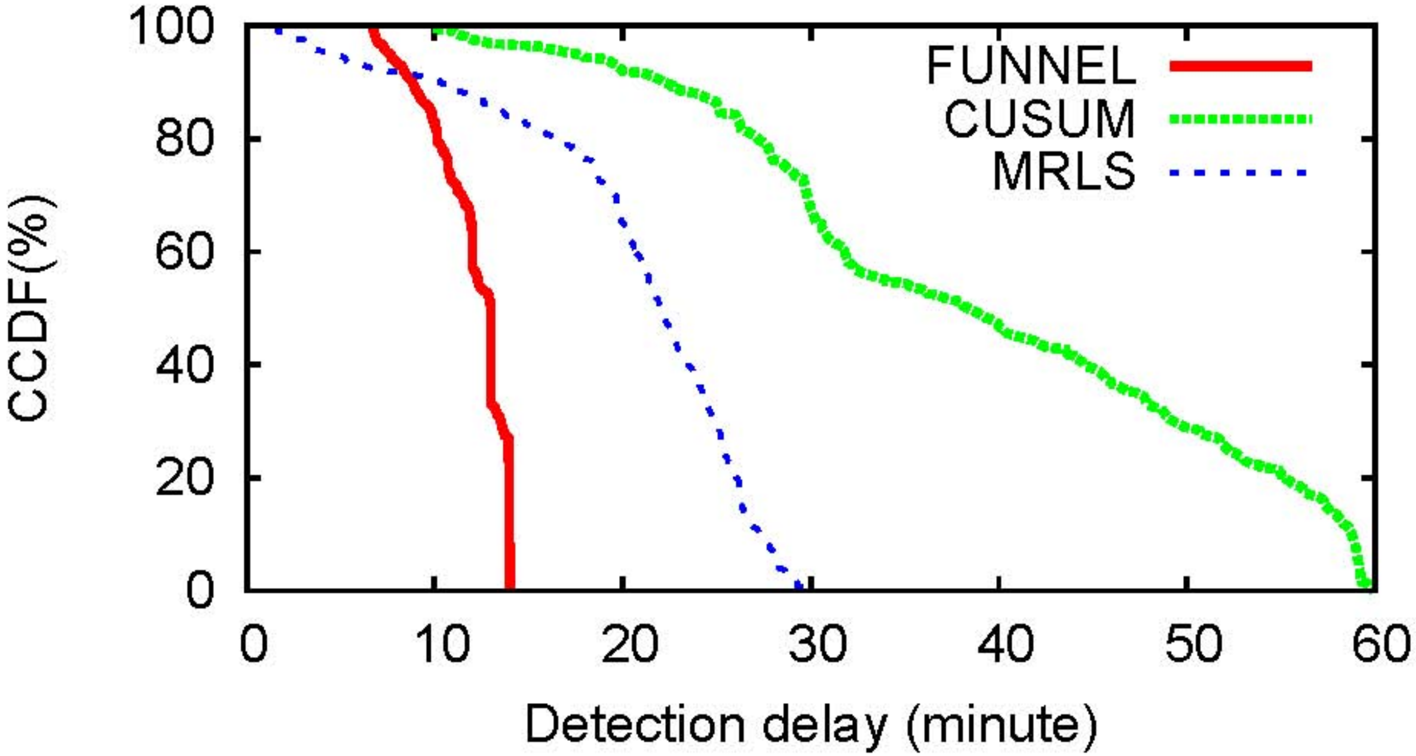
Comparison of Detection Delay

- Detection delay
  - time when a KPI change is detected – time when a KPI change starts



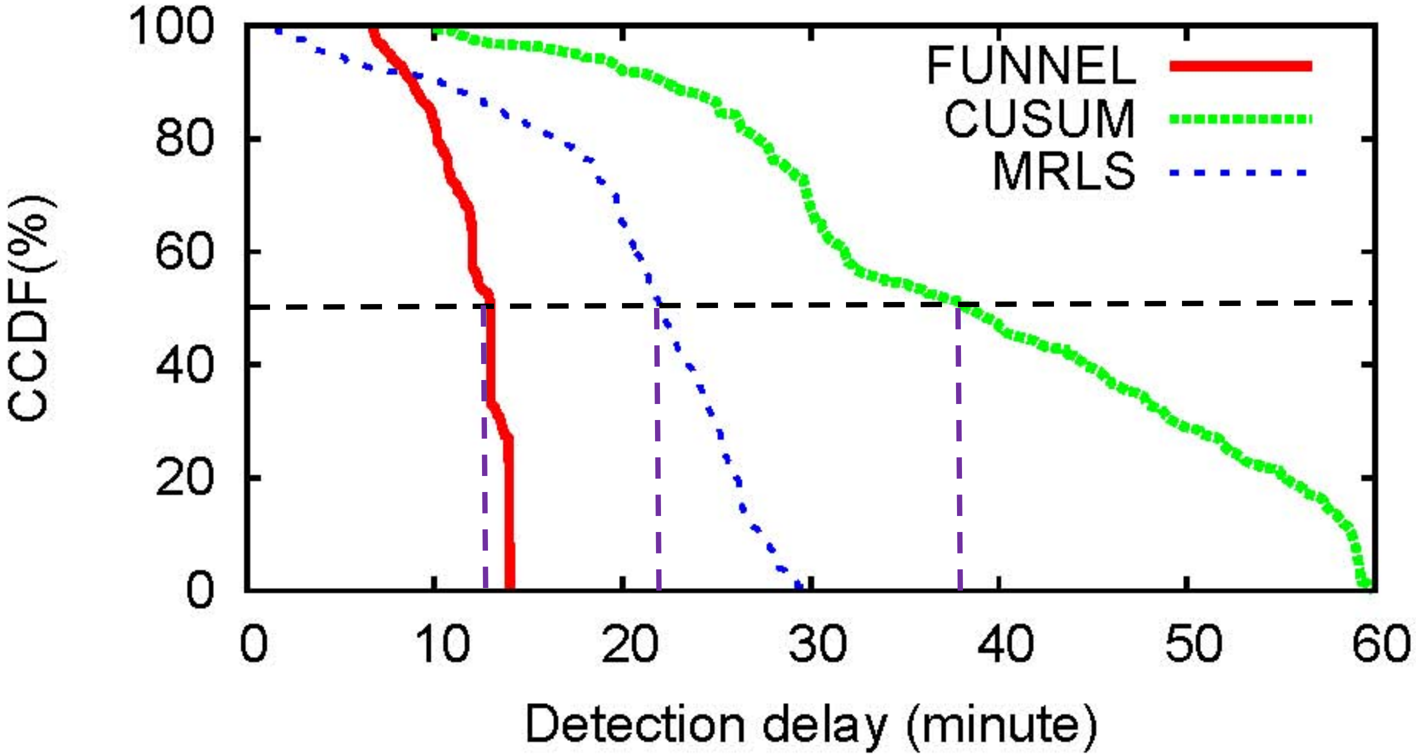
Comparison of Detection Delay

- Comparison results



Comparison of Detection Delay

- Comparison results

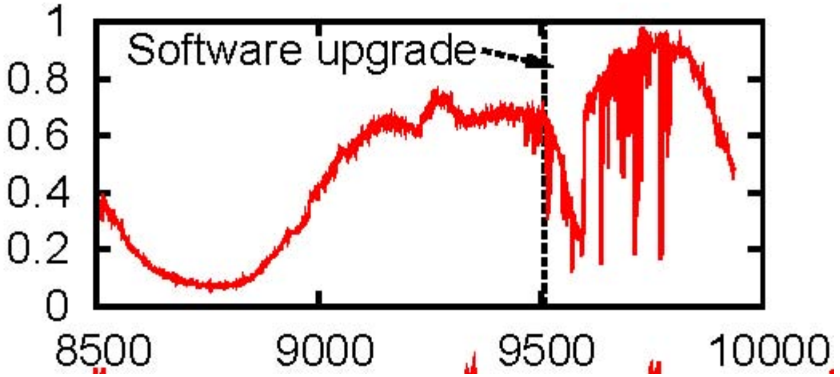


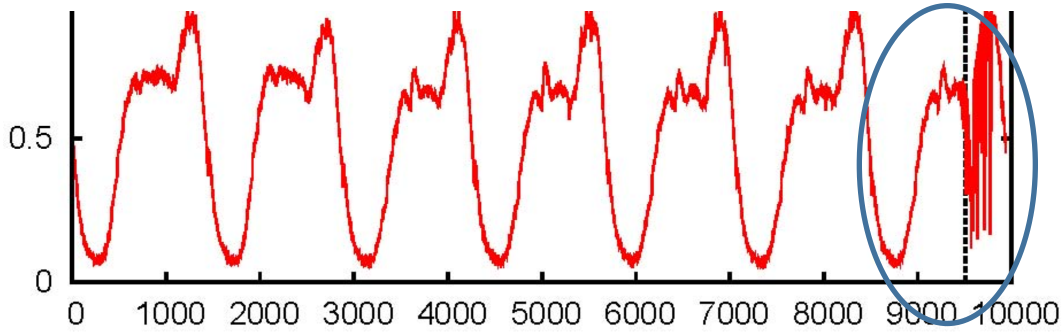
Case Study: An Erroneous Software Upgrade in Advertising

- Methodology
  - A fraction of software changes
  - Not deliver the results to the operators
  - The operators assessed the software changes independently

Case Study: An Erroneous Software Upgrade in Advertising

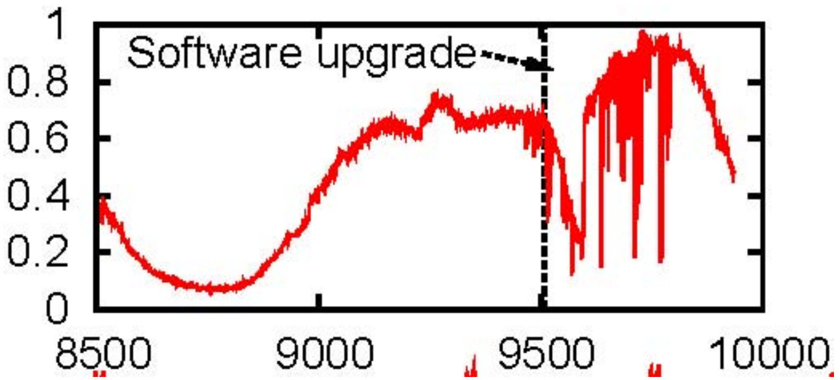
- Methodology
  - A fraction of software changes
  - Not deliver the results to the operators
  - The operators assess software changes independently
- FUNNEL
  - 10 minutes
  - Seasonal KPIs

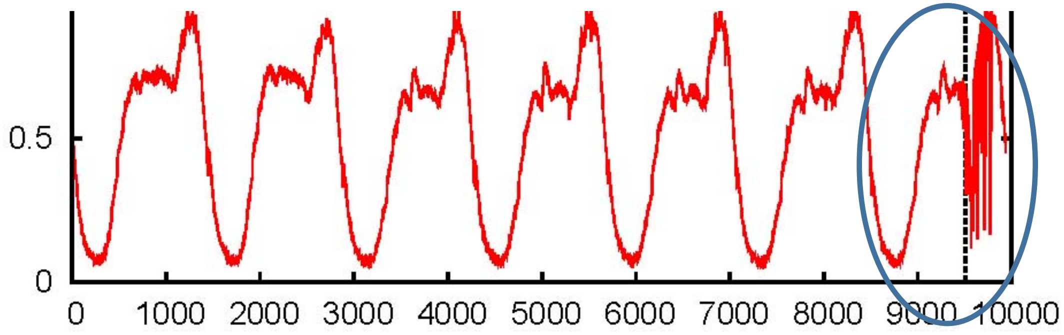


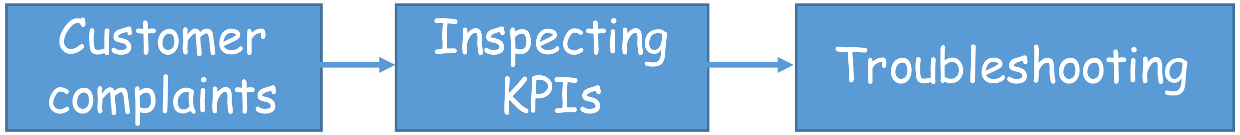


Case Study: An Erroneous Software Upgrade in Advertising

- Methodology
  - A fraction of software changes
  - Not deliver the results to the operators
  - The operators assess software changes independently
- FUNNEL
  - 10 minutes
  - Seasonal KPIs
- The operators
  - 1.5 hours






Customer complaints → Inspecting KPIs → Troubleshooting

Outline

- Background and Motivation
- Challenges
- Key Ideas
- Results
- Conclusion

Conclusion

Challenges of automatic software change impact assessment

- Short detection delay requirement against robustness
- Large number of KPIs
- Diverse types of data
- KPI changes maybe caused by other factors

FUNNEL

- Improved SST – main algorithm contribution of the paper.
- Split testing

Evaluation

- Real-world software changes

Thank you!

zhangsl12@mails.tsinghua.edu.cn