Online experimentation in the cloud

BY M. TOSLALI[1], S. PARTHASARATHY[2], F. OLIVEIRA[2], H. HUANG[2], AND A. K. COSKUN[1]

[1]Boston University; [2]IBM Research

MLSys'22
August 31, 2022

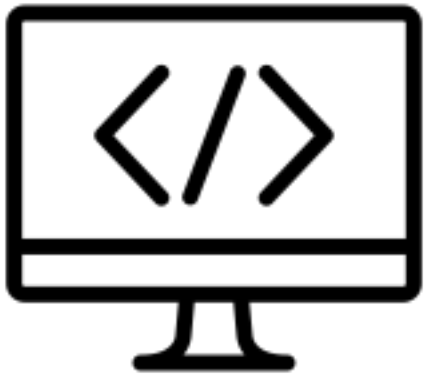
About me

- ***Bio***: 5th year Computer Engineering PhD student at Boston University

- ***Research****: Automated analytics to diagnose performance variations in the code and help prevent them in the frequent code delivery cycles*

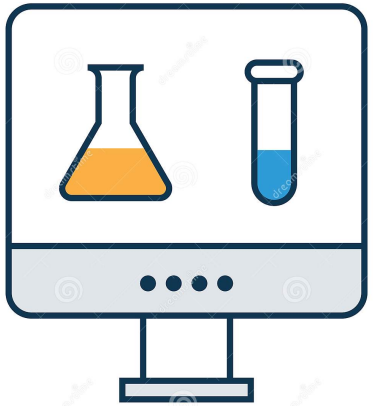
Mert Toslali

This talk in one slide

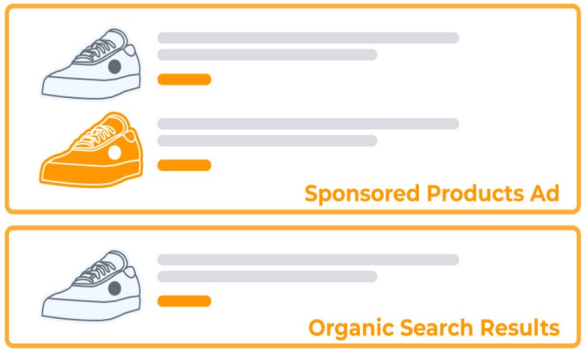


Software

The life of a software



Online experiment

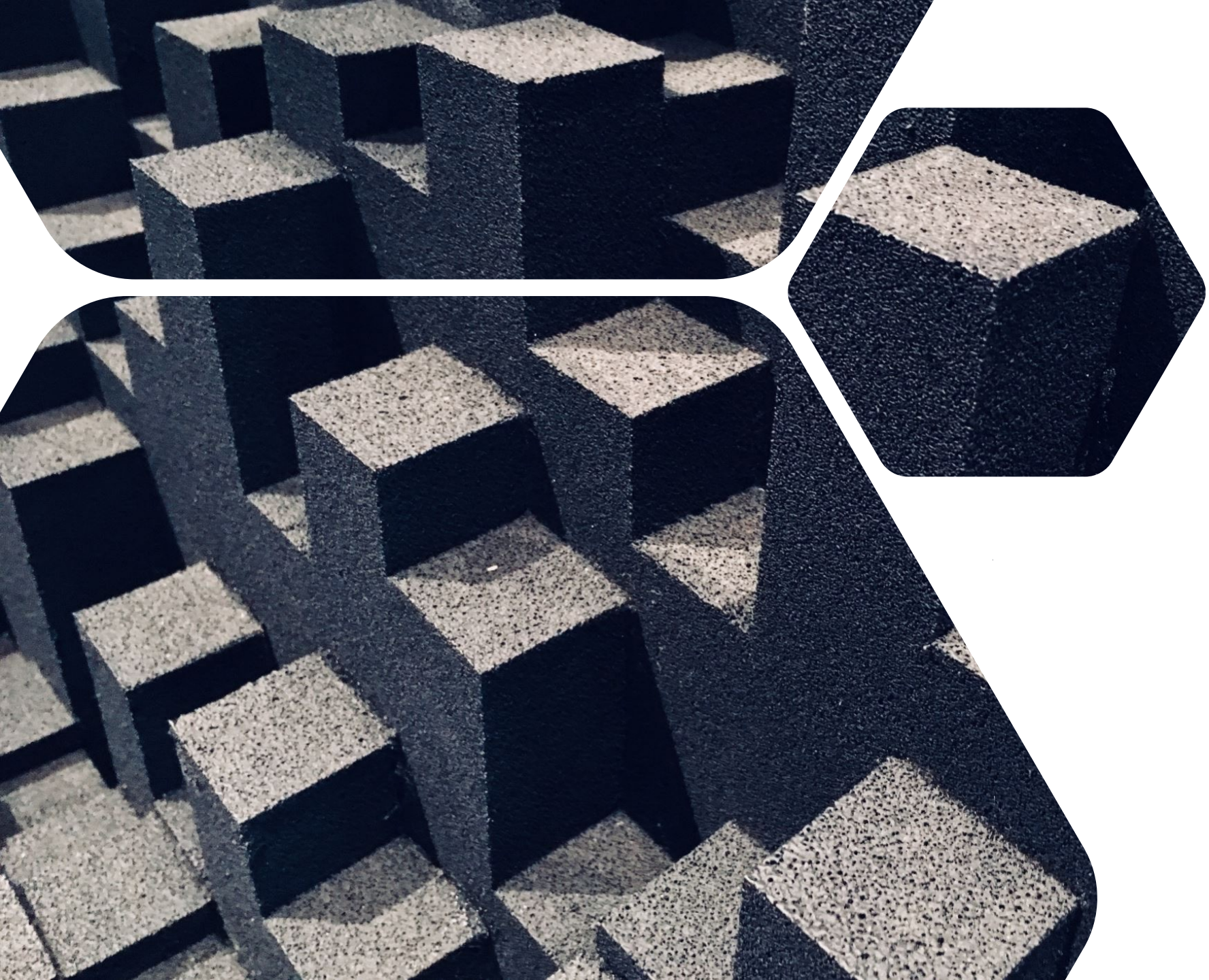


Web/mobile

Cloud

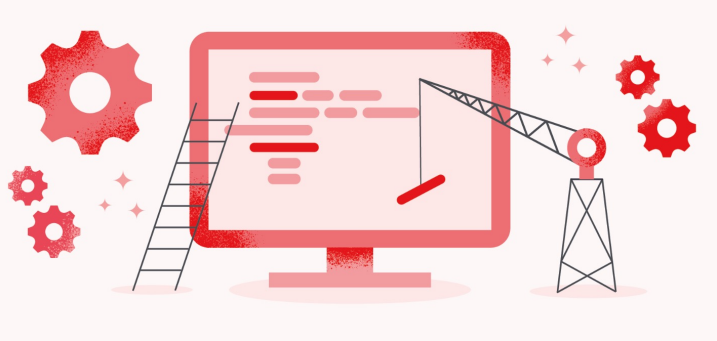
Trustworthy solution

Hands-on



Background

Peace of mind in frequent code delivery?



- Frequent code changes to:
  a) fix problems
  b) satisfying new requirements
  c) …..

- Faster and better software to survive in a digital market

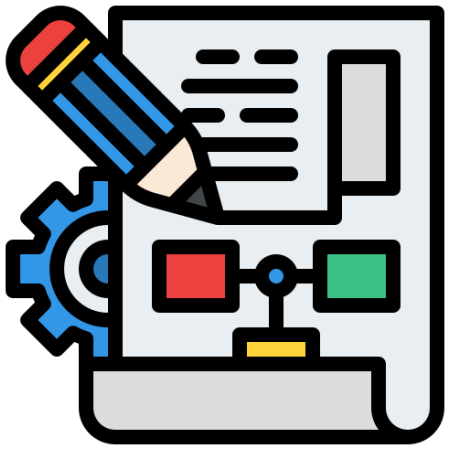
Can one actually have peace of mind when delivering code frequently to the cloud?

The life of a software

- Software development cycle (SDLC)

Analysis & Planning

- Software development cycle (SDLC)

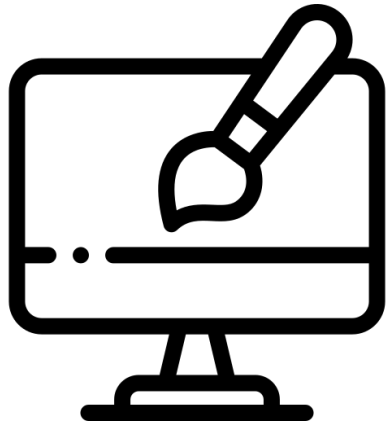


- Plots the scope and purpose of an application
- Set boundaries to keep project's original purpose

Design

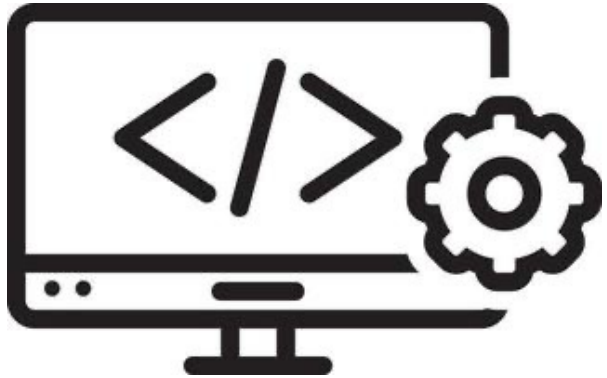
- Software development cycle (SDLC)

- Models the way the software works
- Architecture, user interface, platforms, etc.



Development

- Software development cycle (SDLC)



- Write and develop the software

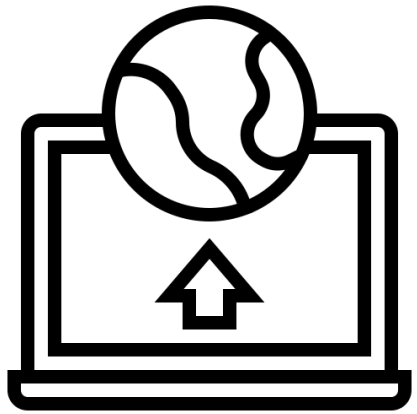
Testing

- Software development cycle (SDLC)

- Report and fix defects, vulnerabilities
- Ensure quality standards

Deployment

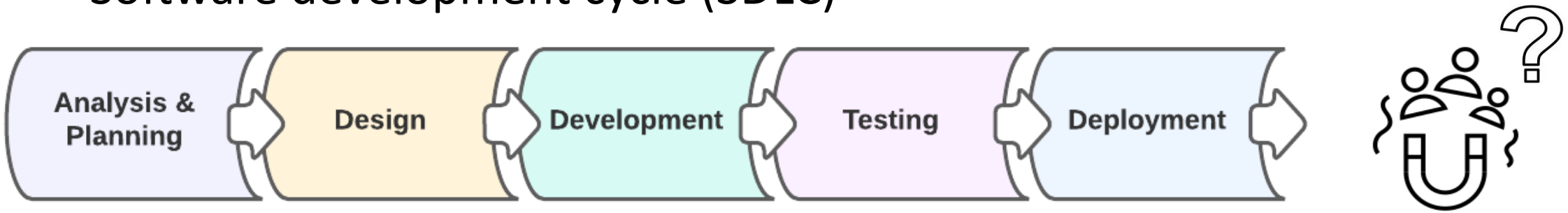
- Software development cycle (SDLC)



- Make software available to users 🚀

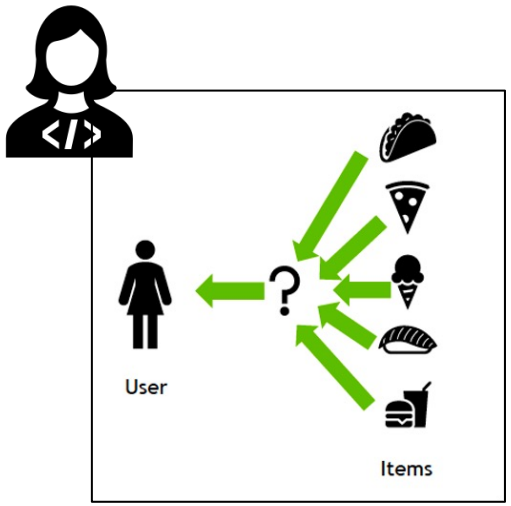
Does your new release deliver value?

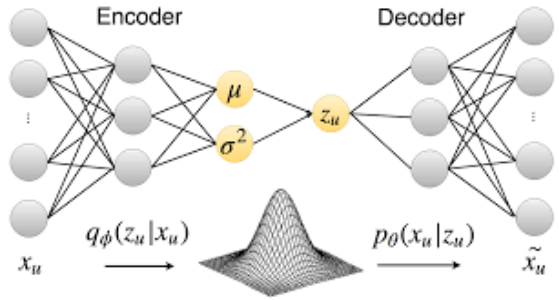
- Software development cycle (SDLC)



How do you know if your new release deliver value to the users/customers?

Does your new release deliver value?





E.g., Autoencoder model for collaborative filtering

Customers first!

Agile model puts customer first!

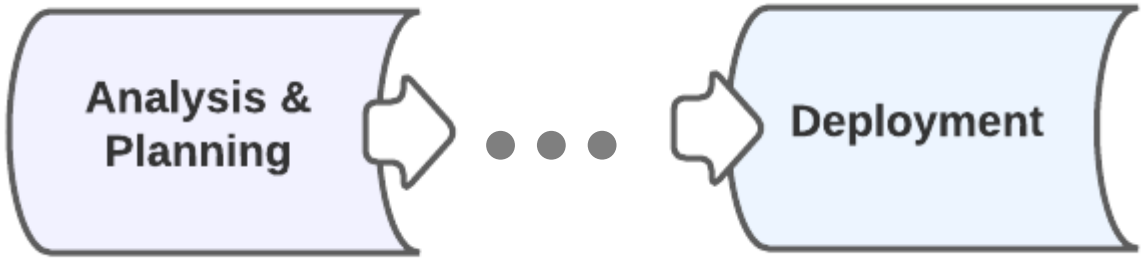
1 Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

2 Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

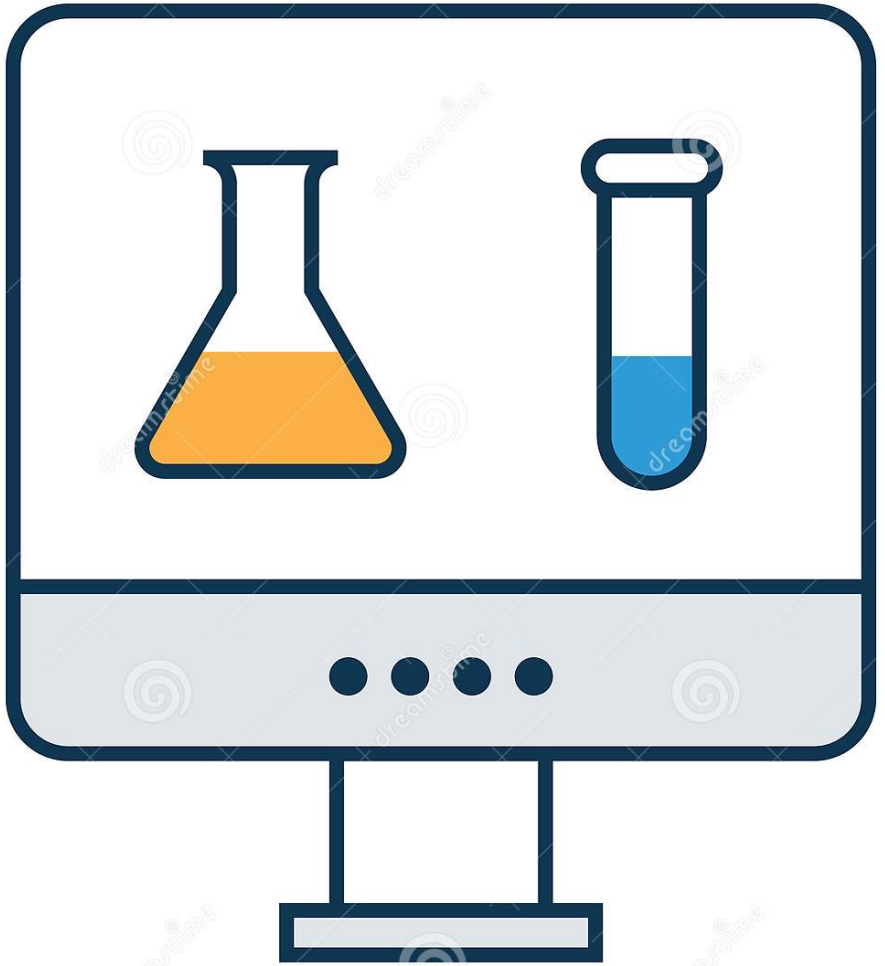
3 Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

The missing piece!

Our beloved developer



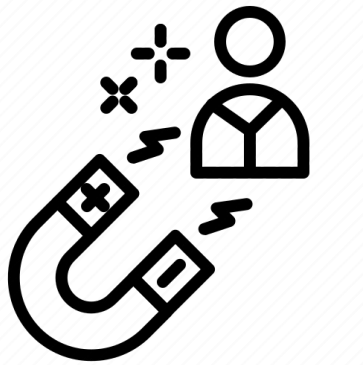
Online experimentation



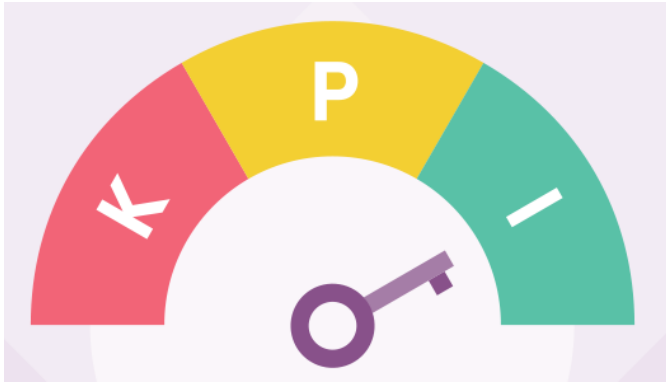
Why is Online Experimentation Necessary?

- You have goals!

Revenue



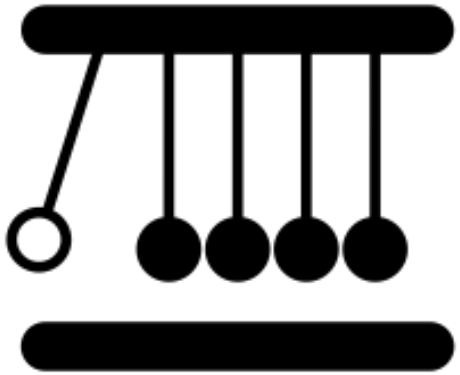
User engagement



- Tracking the progress against goals

Are your actions helping your goals?

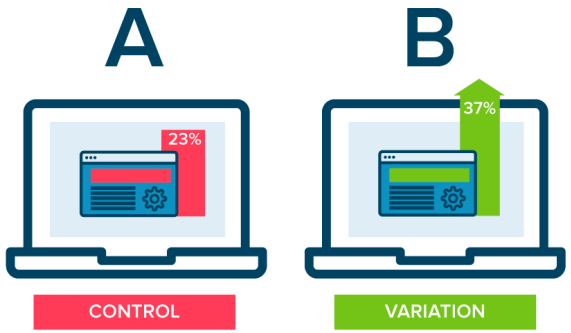
Are your actions helping your goals?



Causality

- Changing the UI:

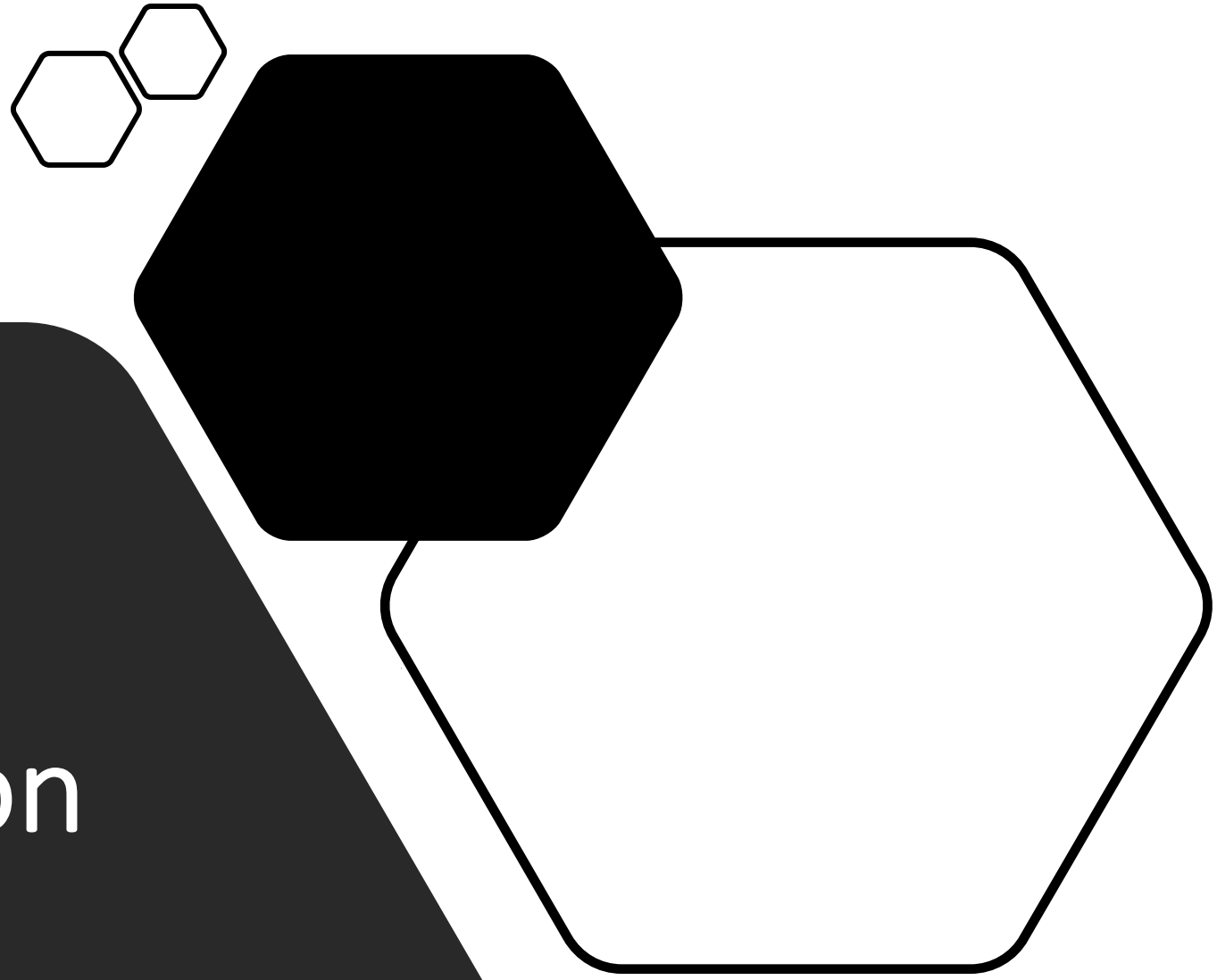
- *How can you confidently know green button produce positive outcomes?*



Online experimentation

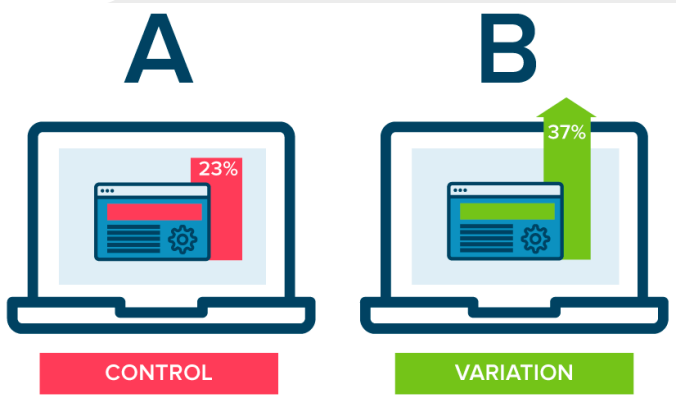
Q/A

Online experimentation in Web/mobile

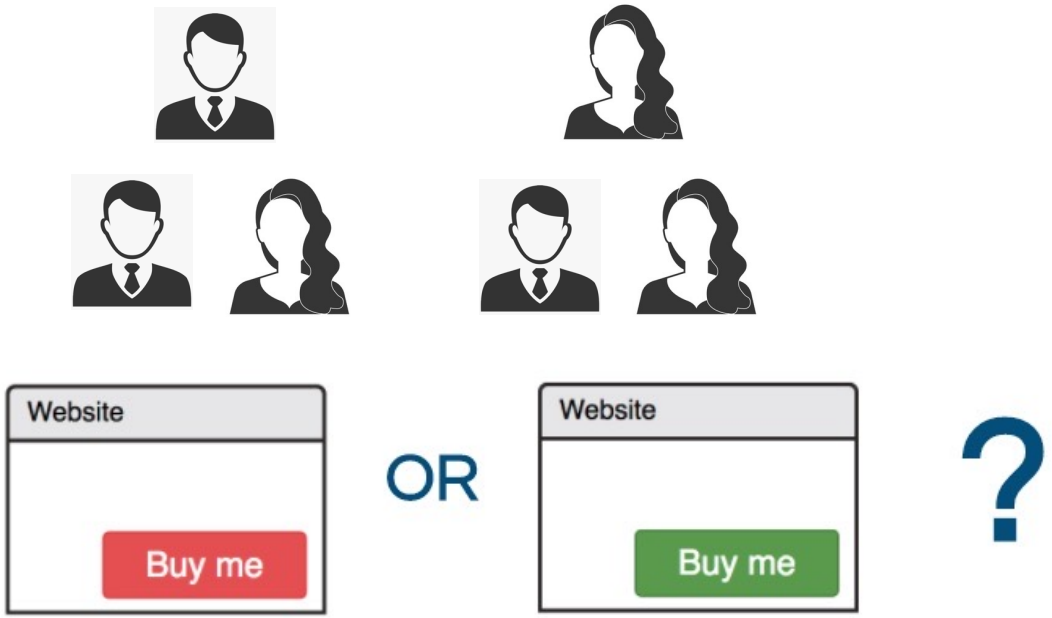


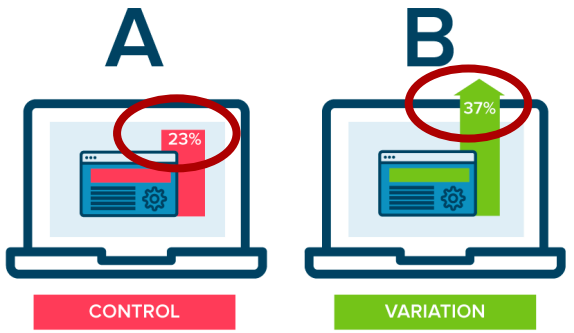
Online experimentation in Web/mobile

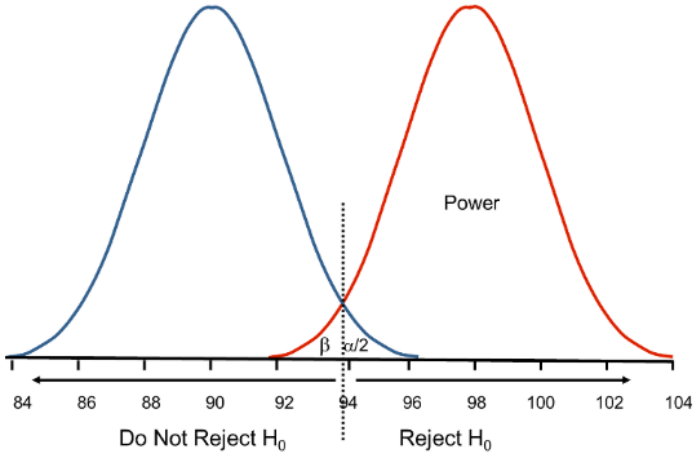
- A/B tests:
  1. deploying two or more competing versions
  2. splitting users across versions
  3. collecting metrics (e.g., user engagement)
  4. determining the best



The green button!







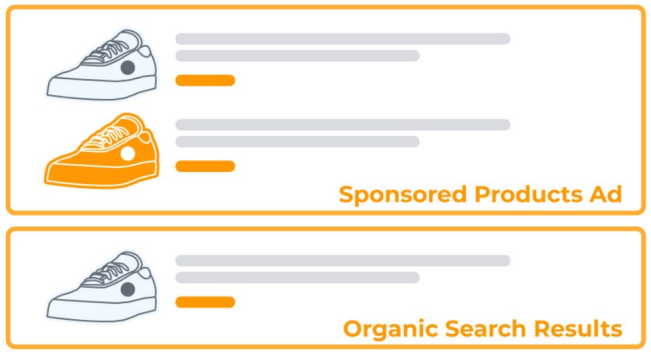
H0: A == B
H1: A != B

confidently go with the new design

Boosting the user engagement!

Which is more effective: option A or option B, C?

Social network advertising



Sponsored search

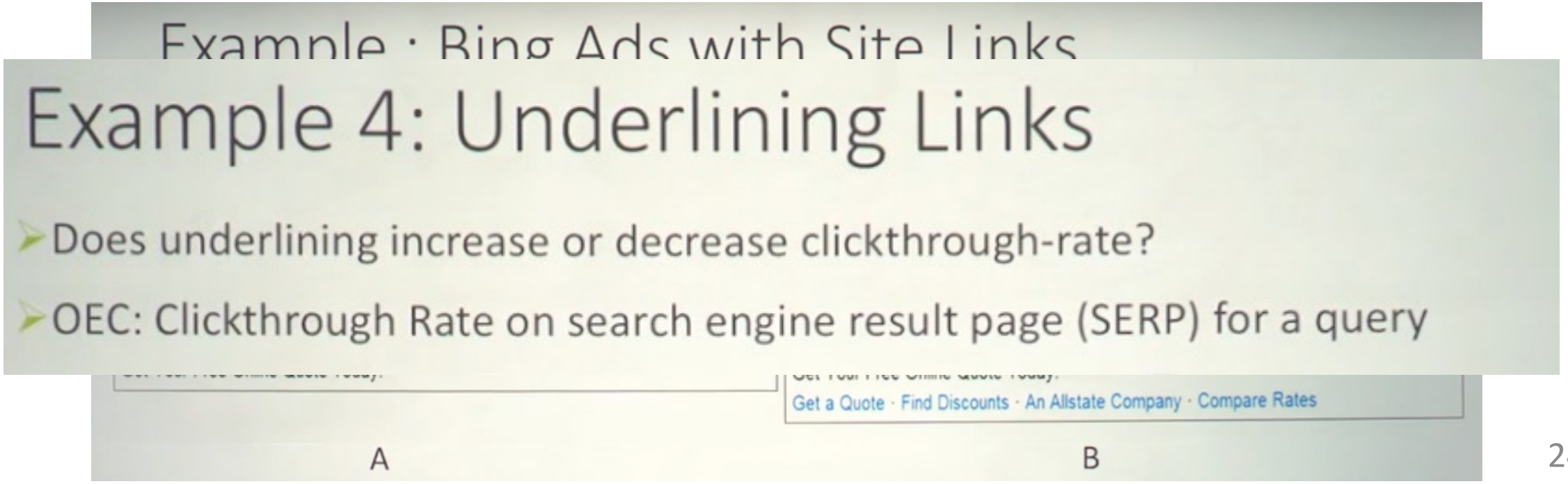
Personalized news

Boosting user engagement on Web/mobile!

- *Sophisticated algorithmic approaches and systems:*
  - *Social network advertising (LinkedIn;* Agarwal et al., WDSM'14*),*
  - *Sponsored search (Microsoft;* Graepel et al., ICML'10*) ,*
  - *Personalized news (Yahoo; Li et al., WWW '10),*
  - *Firebase (firebase.google.com)*
  - *Optimizely (optimizely.com)* ....
- Beyond classical A/B setting → multi-armed bandit heuristics
  - Bayesian algorithms to solve explore vs. exploit trade-offs
  - E.g., show high-CTR ads to the user based on what is already known
- Gains in **revenue** or **CTR** (e.g., 12.5% click lift)

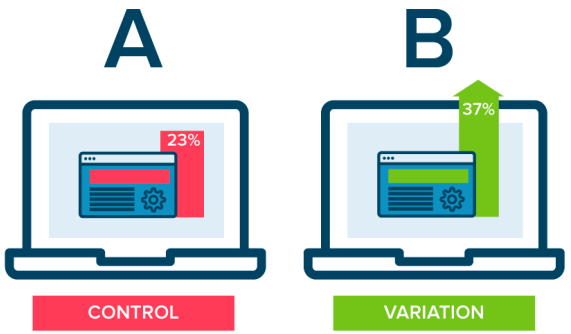
Boosting user engagement on Web/mobile!

- A/B Testing Pitfalls, Ronny Kohavi (CXL 2016)
  - *"There is no single Bing!"*
  - 100k – 10M of users participate in experiments (90% of all users)



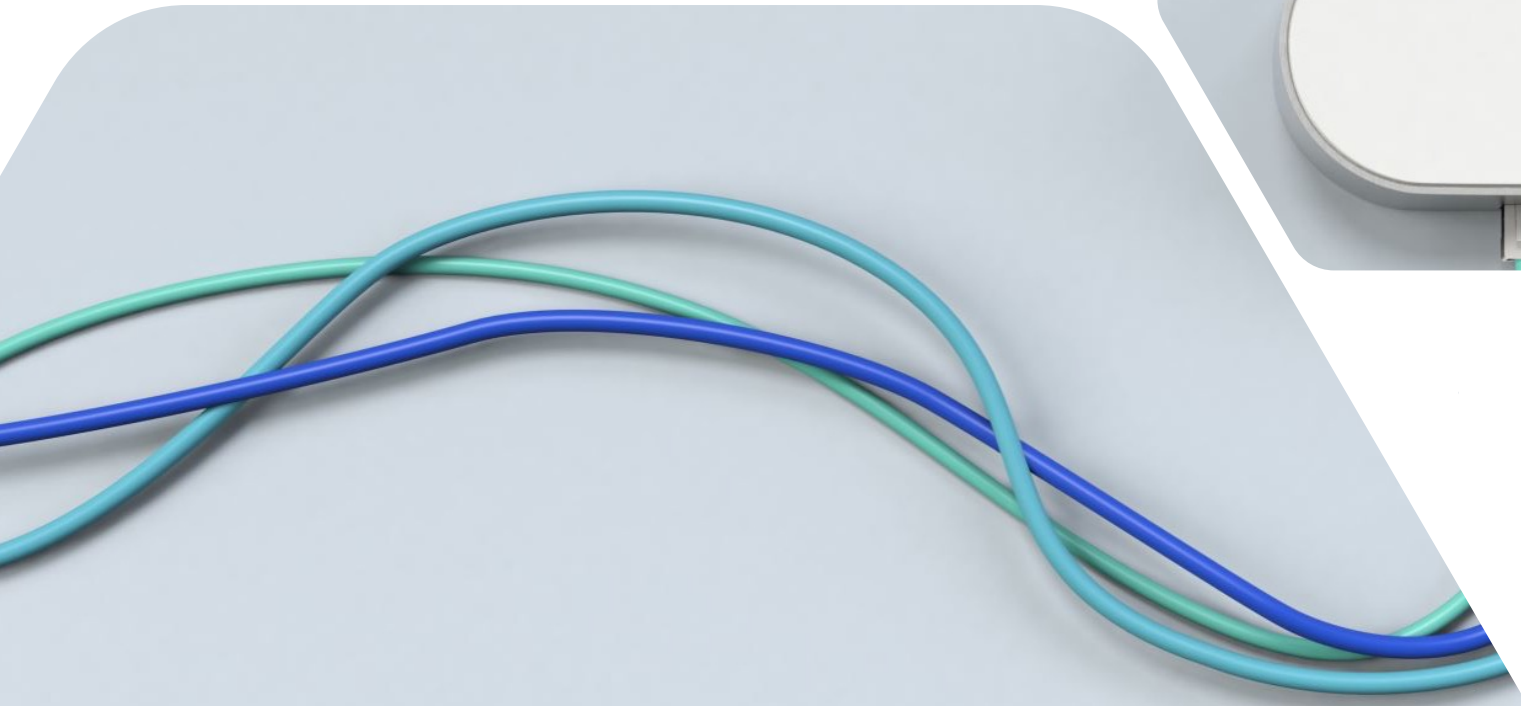
What about cloud?

- Rich landscape of research and tools are available in Web/mobile



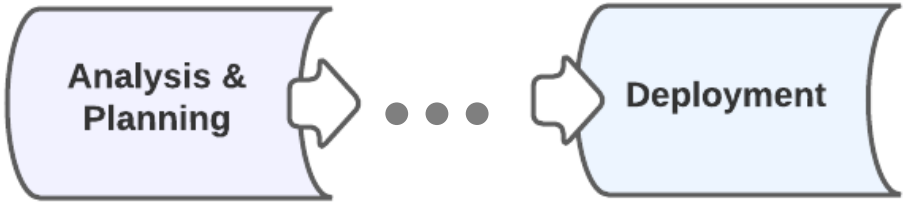
In cloud, business results conflate with operational concerns

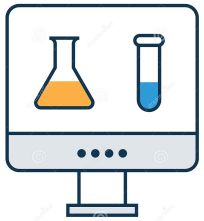
Q/A

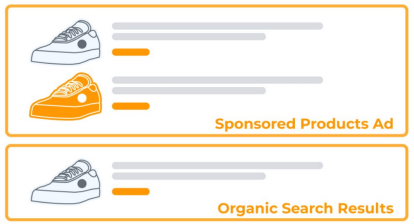


Online experimentation in the cloud

The cloud setting







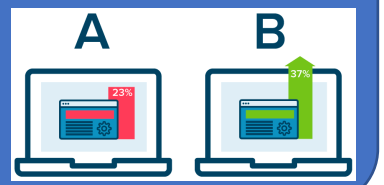
Rethinking online experimentation in the cloud

Factor in the SLOs

Traditional A/B testing in Web/mobile

Compare 2+ versions and identify best one

- User engagement
- Click through rate
- Revenue



Service level objectives; SLOs

100ms delay = ⇩ 1% sales

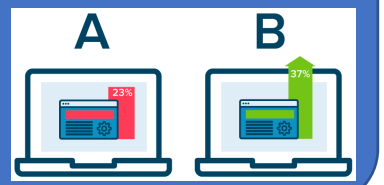
500ms delay = ⇩ 20% traffic

Challenging Cloud APIs

Traditional A/B testing in Web/mobile

Compare 2+ versions and identify best one

- User engagement
- Click through rate
- Revenue



Service level objectives; SLOs

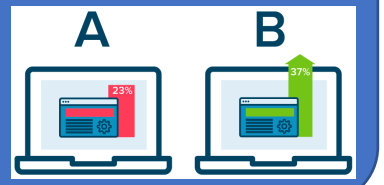
Programmatically split users via APIs

Validation experiments

Traditional A/B testing in Web/mobile

Compare 2+ versions and identify best one

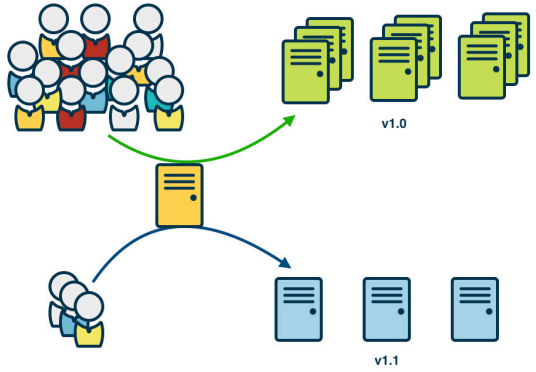
- User engagement
- Click through rate
- Revenue



Service level objectives; SLOs

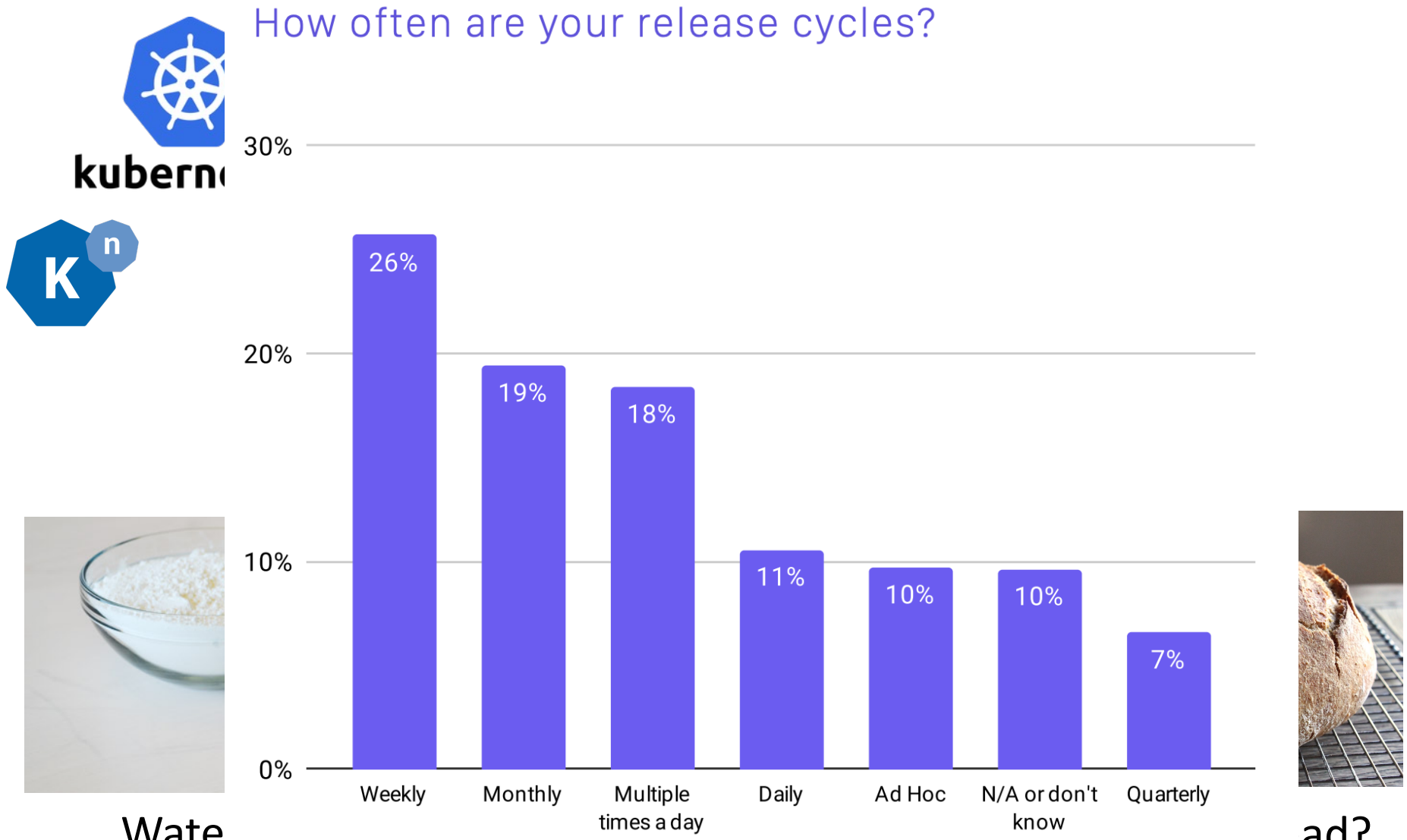
Programmatically split users via APIs

Validation experiments!



Canary release
Conformance tests
Dark-launch
….

Online experimentation in the cloud



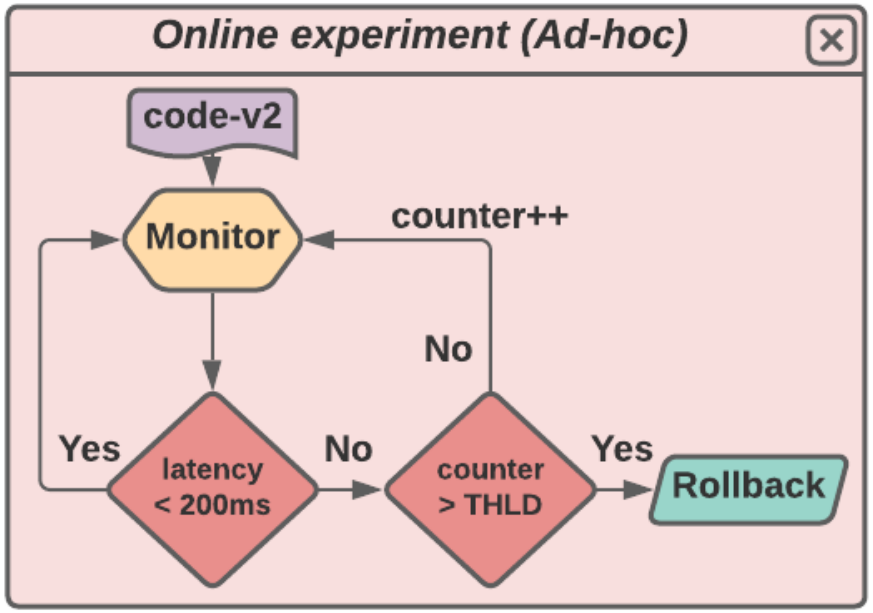
Wate ad?

An art of cloud-native code delivery

- Agile practices are analytics problems at a fundamental level!
  - Comparing competing versions
  - Factoring in the SLOs
  - Adjusting user traffic split/segmentation
  - Confidently promoting the best version!

- *Cloud-native automation solutions focus on the narrow problem of progressive rollout of a new application version*

Automation solutions



Ad-hoc checks on metrics to shift user traffic

flagger.app

argoproj.github.io/argo-rollouts

Solely relying on ad-hoc checks is too simplistic and fragile!

Q/A

Principles for a trustworthy solution

Principles of a trustworthy solution

- Practitioners lack proper solutions to code releases methodically!
  - Web/mobile solutions are not applicable
  - Cloud-native solutions provide ad-hoc mechanisms

- *Need to rethink online experimentation for the cloud era, study it, and provide a practical solution to this timely problem*

Data-driven!

Data-driven, statistically rigorous approach is essential

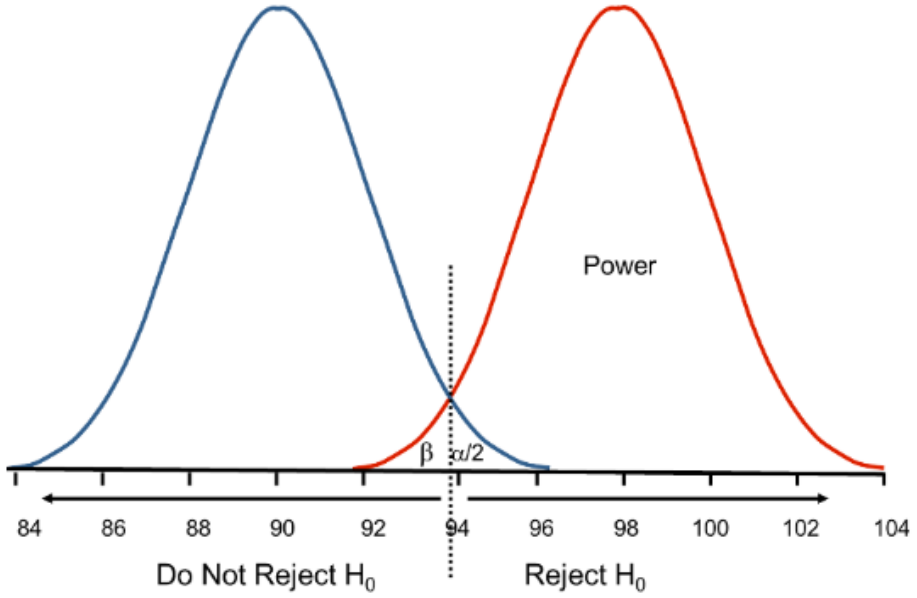
Accuracy

- Resemble the oracle
- Rollout must result in the correct answer

Repeatability

- Same outcome must occur every time

Statistical rigor!



Statistical rigor

- Consider all observations
- No premature decisions
- Accurate assessment of versions
- Resiliency to high metric variance

Convolving business and operational concerns

SLOs (Service-level Objectives)

- Performance or correctness metrics
  - Mean latency, error rate

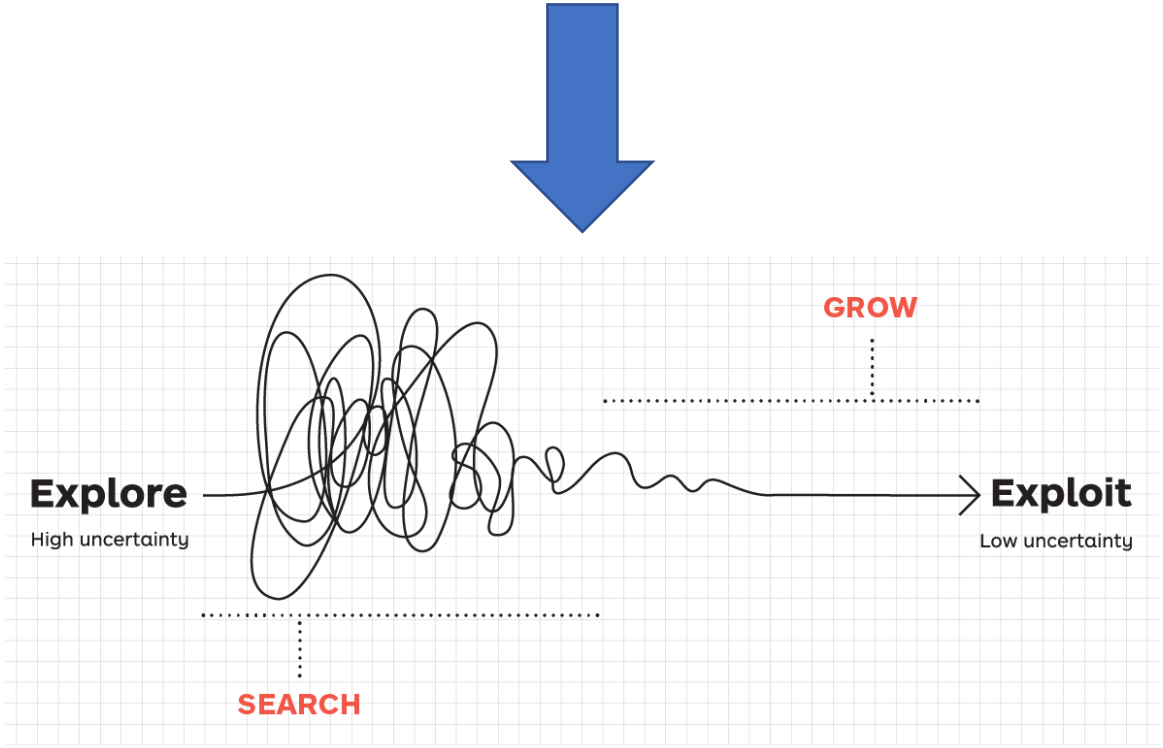
Business rewards

- User engagement or monetary metrics
  - Conversion rate, revenue etc.

Among the versions that have acceptable performance, which one benefits the business the most?

Deliver business results as you roll out!

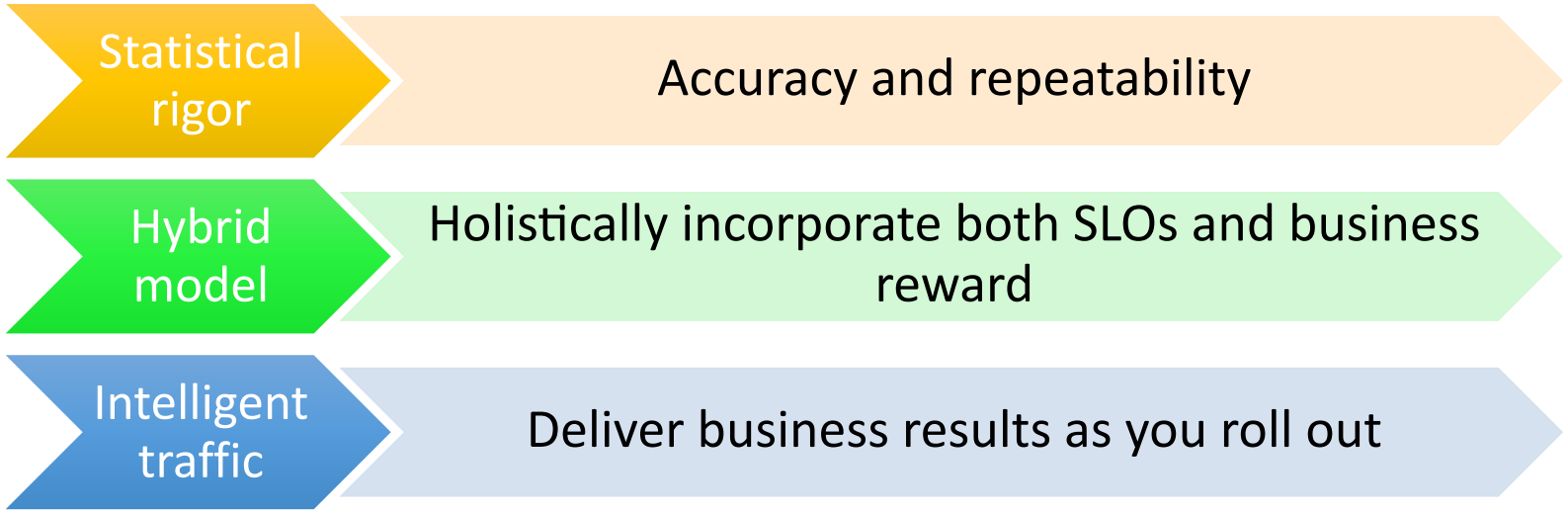
Among the versions that have acceptable performance, which one benefits the business the most?



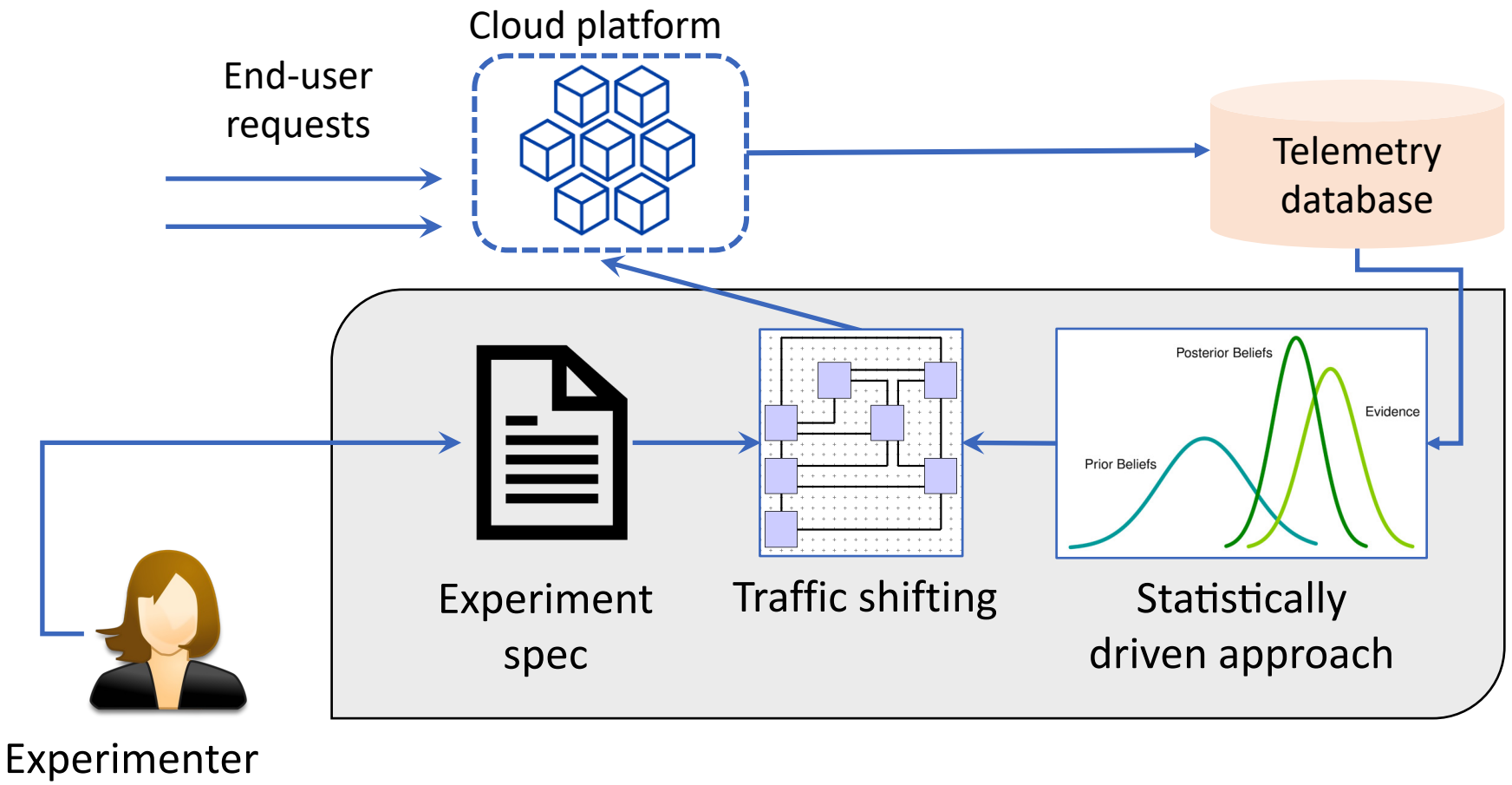
Q/A

A prospective solution

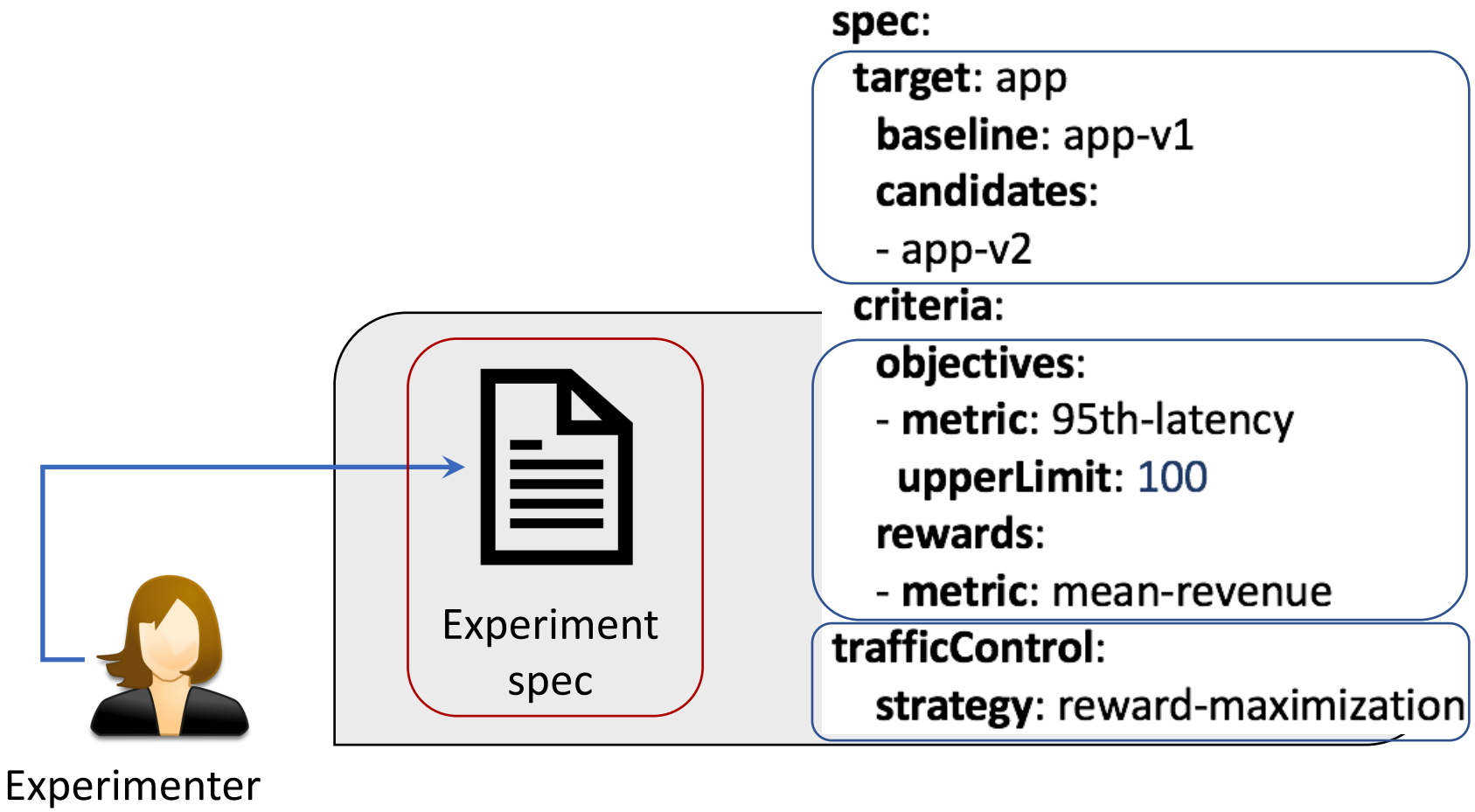
Principles summarized



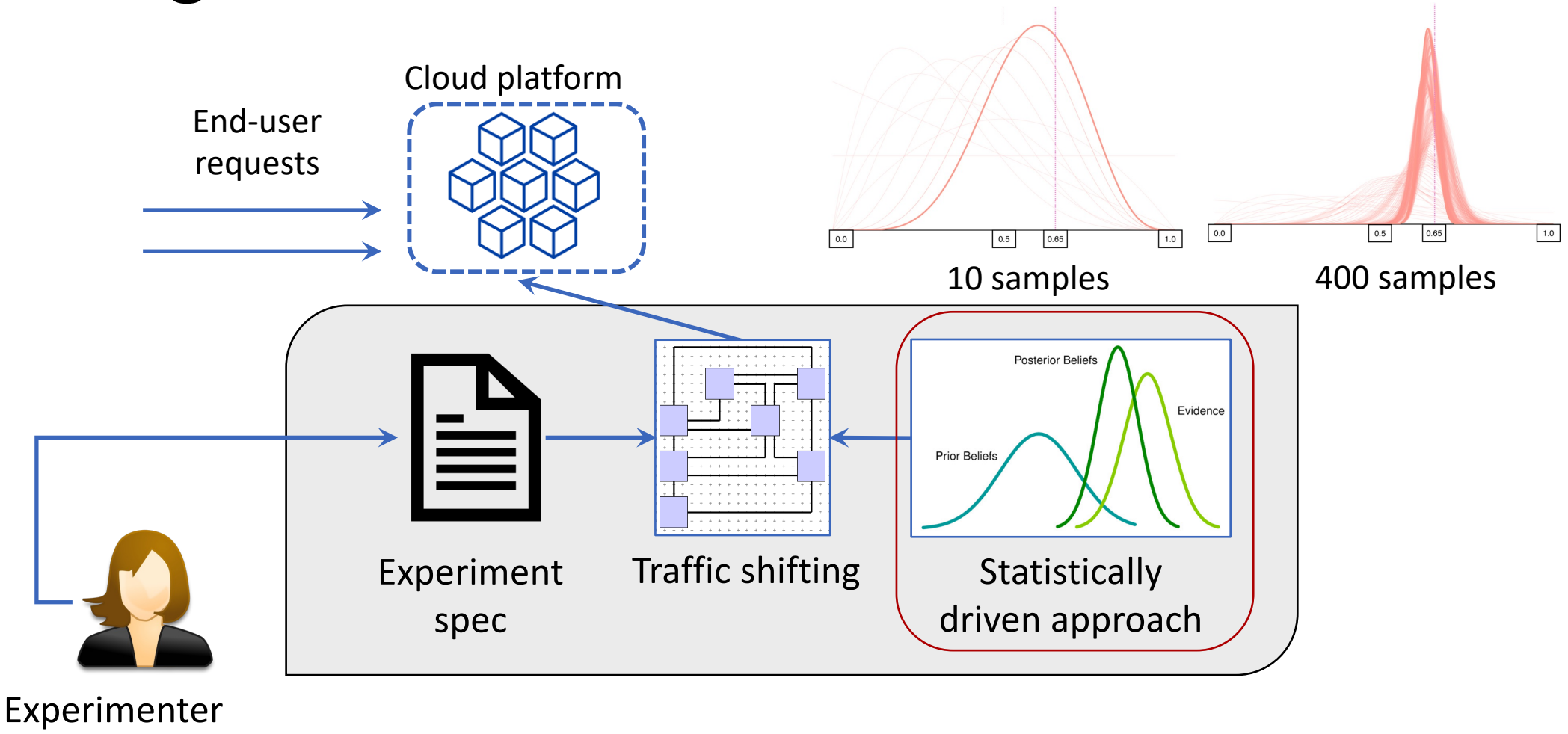
A prospective solution



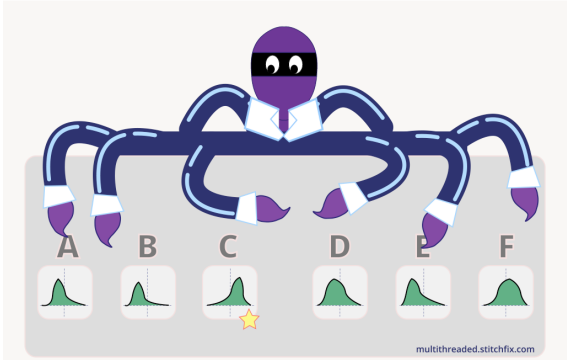
Experiment spec



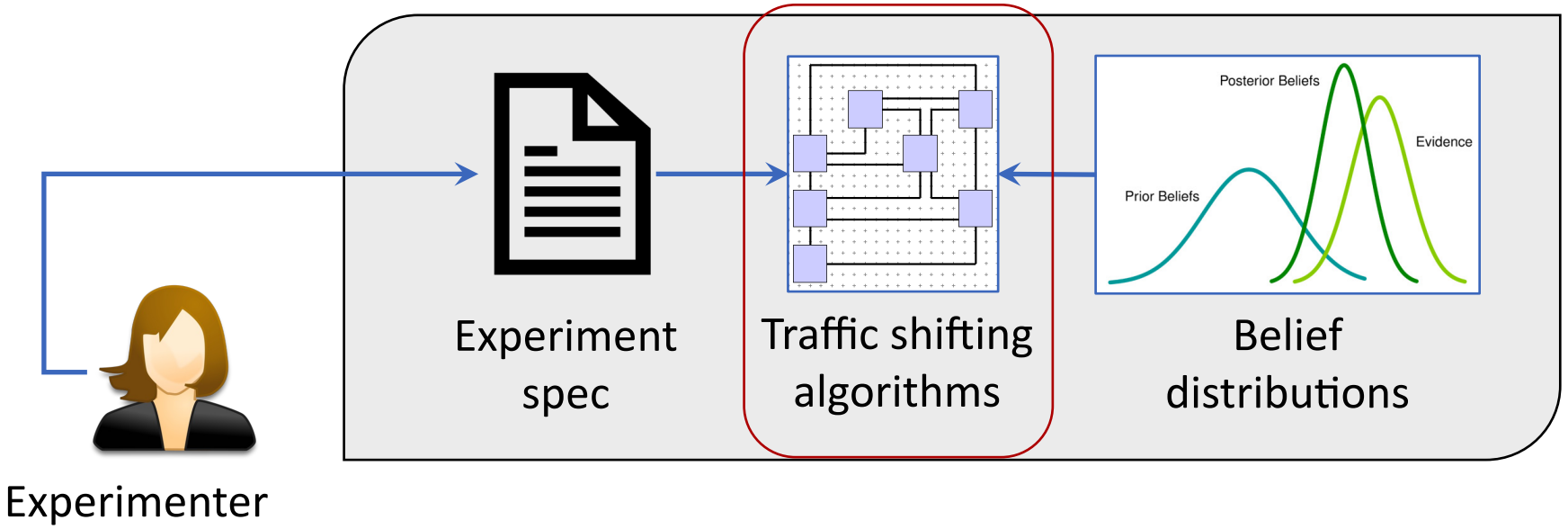
Learning…


Cloud platform
End-user requests
0.0
0.5
0.65
1.0
10 samples
400 samples
Posterior Beliefs
Evidence
Prior Beliefs
Experiment spec
Traffic shifting
Statistically driven approach
Experimenter

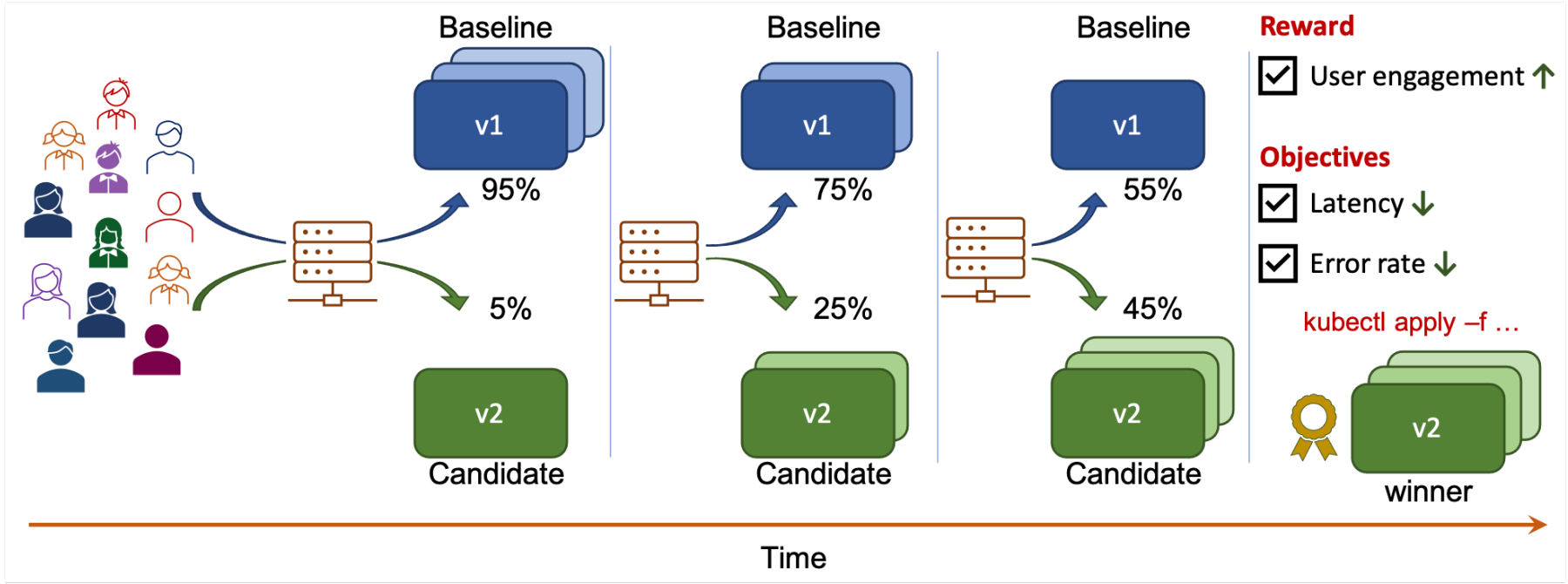
Adjusting the traffic



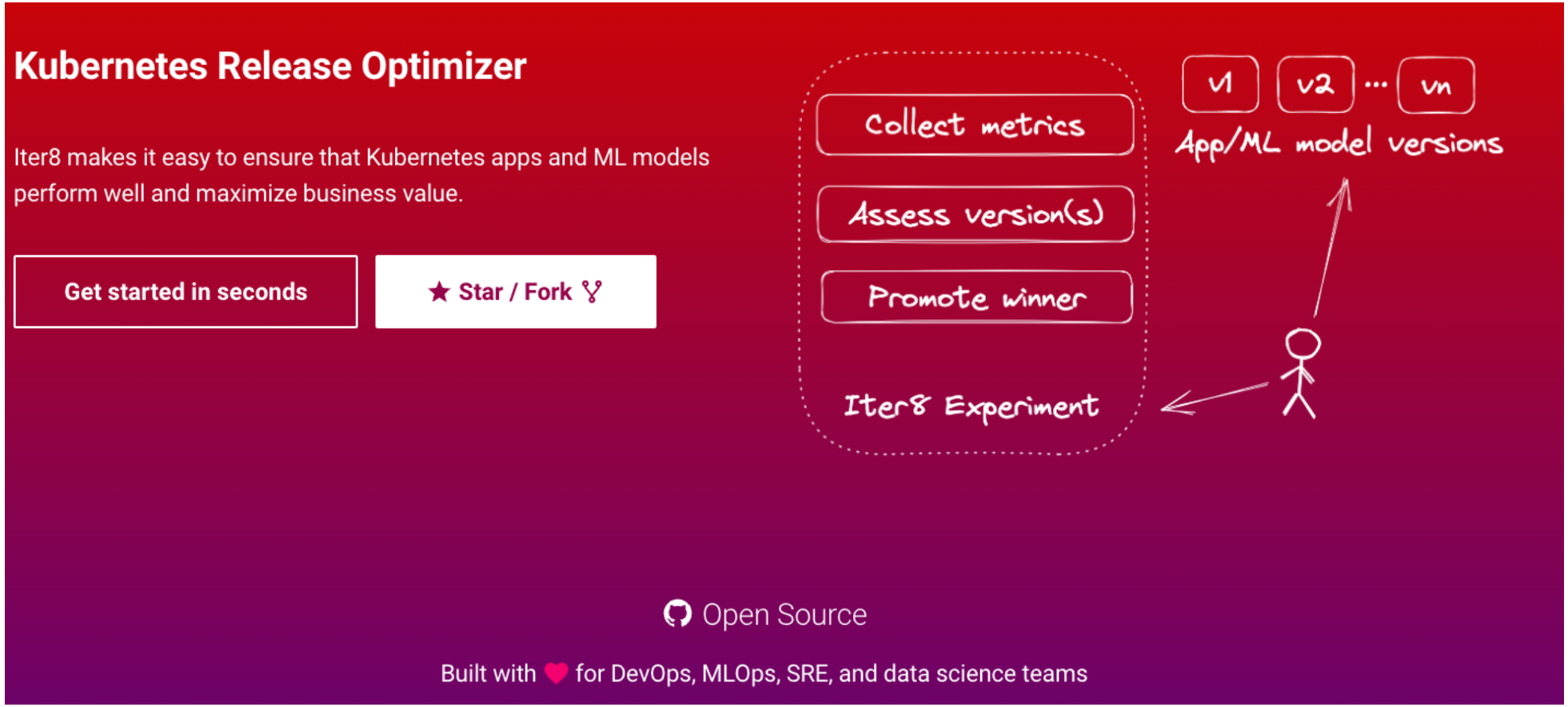
Trade-off: exploration vs. exploitation



An executive example!



Iter8

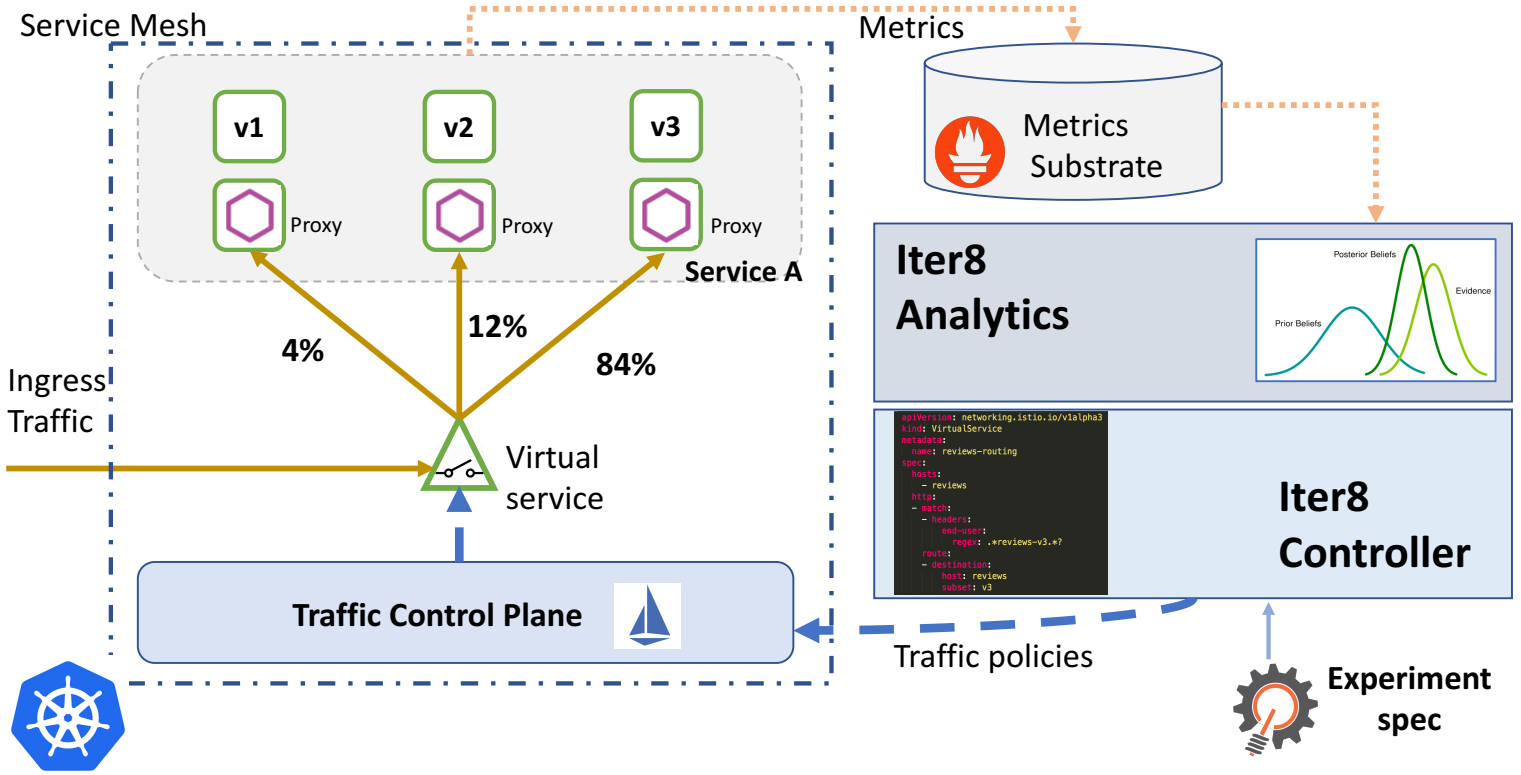


Website: https://iter8.tools/

Research paper: M. Toslali, S. Parthasarathy, F. Oliveira, H. Huang, and A. Coskun***. Iter8: Online Experimentation in the Cloud***. In Proceedings of the ACM Symposium on Cloud Computing (SoCC '21). Association for Computing Machinery; https://doi.org/10.1145/3472883.3486984

Reach out to me at toslali@bu.edu

Iter8 discussion



kubernetes

Istio KServe SELDON LINKERD

Iter8 discussion

CONTROLLER

- Orchestration and API utilization

ANALYTICS

- Mathematical framework

Online Bayesian learning

Iter8 discussion

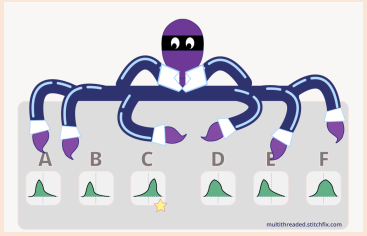
CONTROLLER

- Orchestration and API utilization

ANALYTICS

- Mathematical framework

Online Bayesian learning



Multi-armed bandit algorithms

PBR
- Exploit the best
- Maximize reward!

PBR-split
- Exploit top-2
- Maximize confidence!

PBR: Posterior Bayesian Routing

Iter8

- In our SoCC'21 paper, we introduce Iter8, a system for online experimentation of cloud applications
- Iter8's accuracy in the context of canary releases
  - Iter8 results in correct outcomes (93%), outperforming ad-hoc approaches
- Iter8's ability to maximize business reward
  - Iter8 outperforms alternatives in

    a) maximizing reward

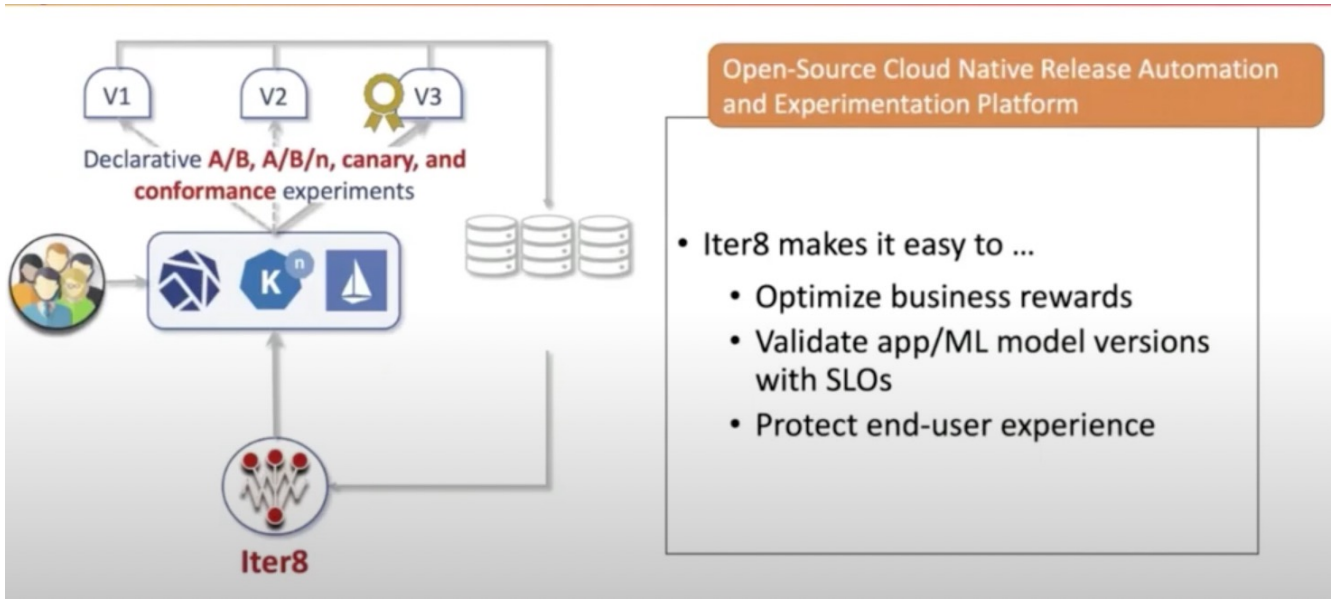
    b) keeping user traffic to the optimal version

    c) finding the best version with significantly fewer requests

Q/A

Hands-on

Hands-on background



Iter8: Online experimentation for Cloud!

- *"Kubernetes is the leading infrastructure, accounting for 83% of the market"*

kubernetes

Platform Setup

1. Create Kubernetes cluster
   - Local cluster using minikube

```
minikube start --cpus 8 --memory 12288
```

2. Clone Iter8 repo

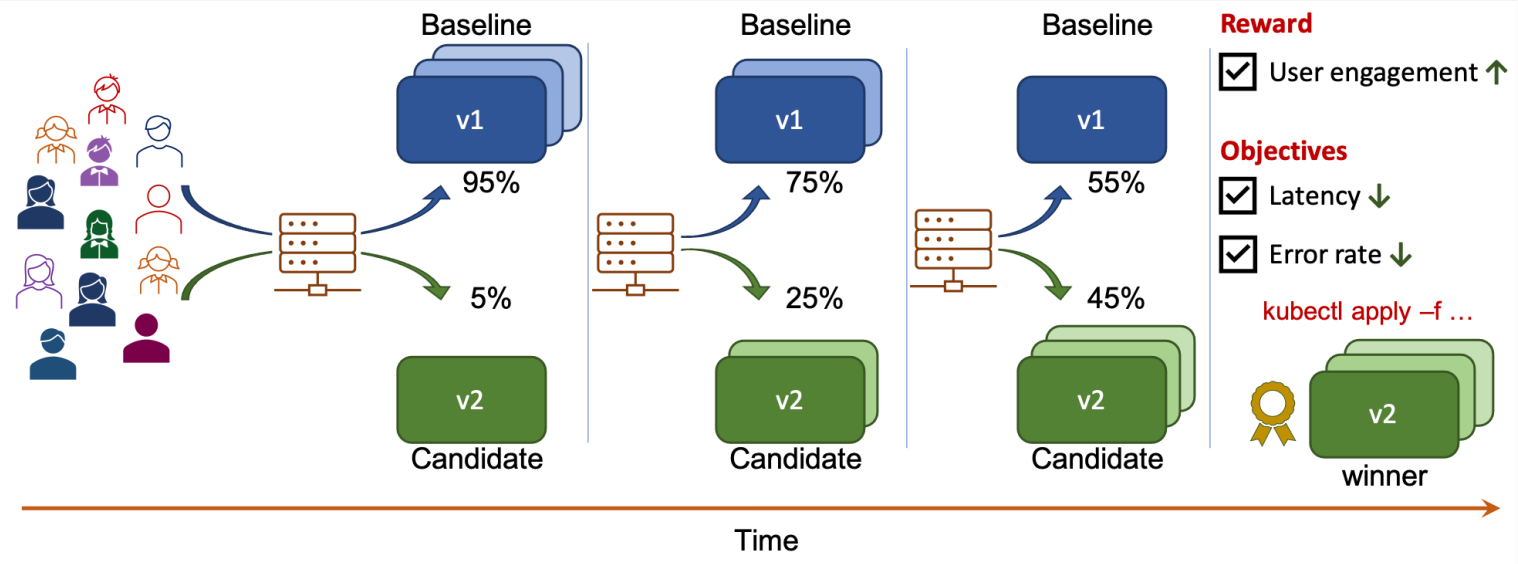
```
git clone https://github.com/iter8-tools/iter8.git
cd iter8
export ITER8=$(pwd)
```

3. Install Istio, Iter8 and Prometheus

```
$ITER8/samples/istio/quickstart/platformsetup.sh
```

Hybrid (A/B + SLOs) testing

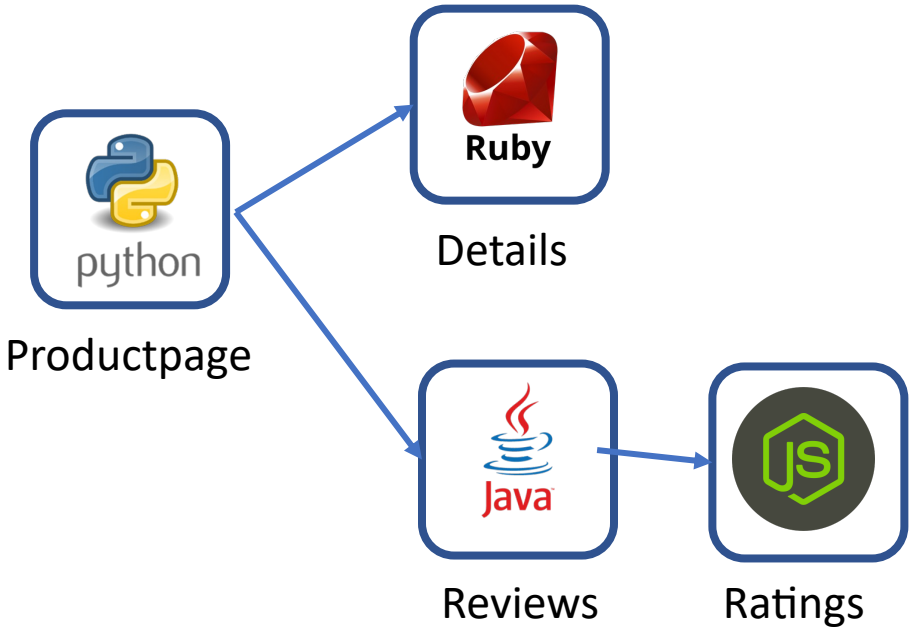
1. SLOs: latency and error rate
2. Define reward
3. Use Prometheus as the provider
4. Combine SLO validation with progressive traffic shifting



https://iter8-tools.github.io/iter8/0.7/tutorials/istio/quick-start/

Hybrid (A/B + SLOs) testing

1. Install bookinfo application
2. Generate requests to your app
3. Define metrics
4. Launch an experiment

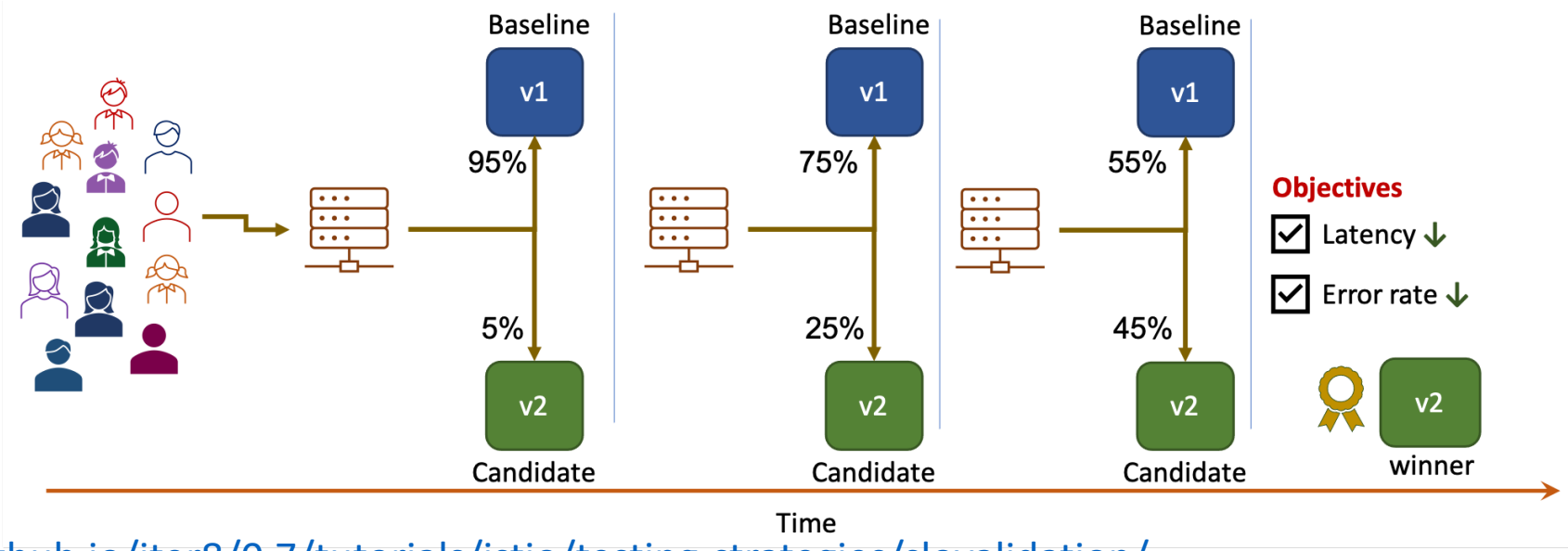


Bookinfo sample application composed of four separate microservices

https://iter8-tools.github.io/iter8/0.7/tutorials/istio/quick-start/
https://github.com/iter8-tools/iter8/blob/v0.7/samples/istio/quickstart/experiment.yaml
https://github.com/iter8-tools/iter8/blob/v0.7/samples/istio/quickstart/productpage-v2.yaml

SLO Validation

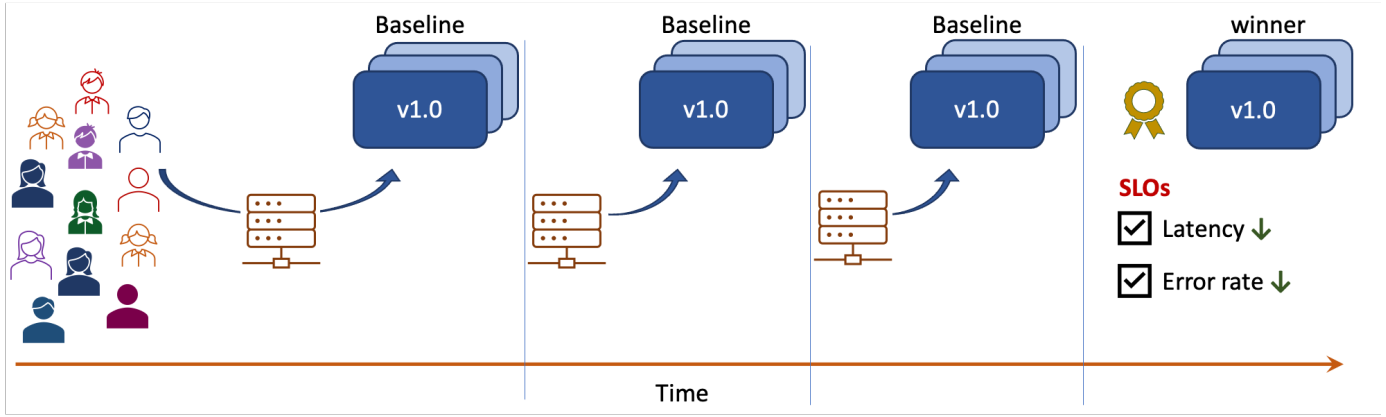
1. SLOs: latency and error rate
2. Use Prometheus as the provider
3. Combine SLO validation with progressive traffic shifting



https://iter8-tools.github.io/iter8/0.7/tutorials/istio/testing-strategies/slovalidation/

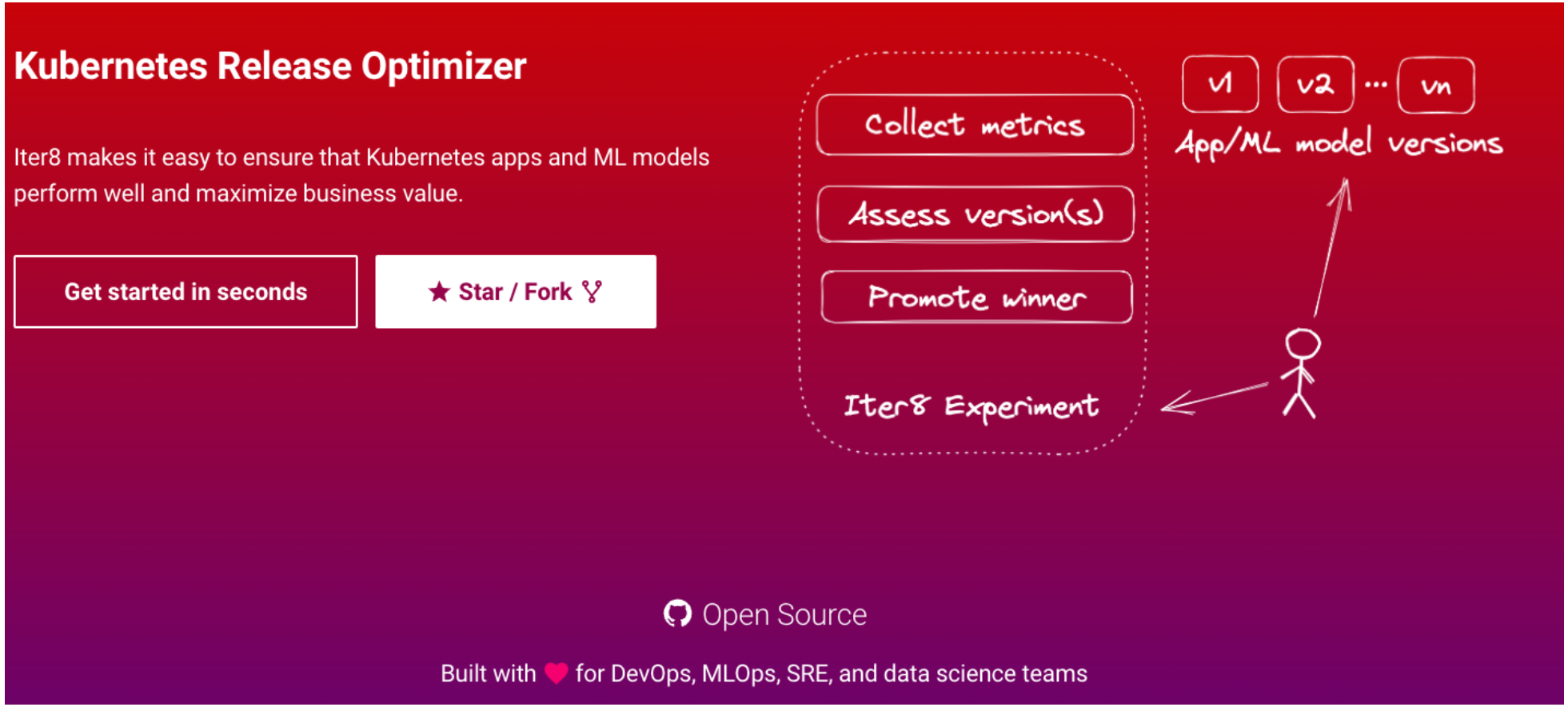
Conformance test

1. SLOs: latency and error rate
2. Use Prometheus as the provider



https://iter8-tools.github.io/iter8/0.7/tutorials/istio/testing-strategies/conformance/

Questions



Website: https://iter8.tools/

Research paper: M. Toslali, S. Parthasarathy, F. Oliveira, H. Huang, and A. Coskun***. Iter8: Online Experimentation in the Cloud***. In Proceedings of the ACM Symposium on Cloud Computing (SoCC '21). Association for Computing Machinery; https://doi.org/10.1145/3472883.3486984

Reach out to me at toslali@bu.edu