TOWARD A SYSTEM-WIDE AND INTERDISCIPLINARY PERSPECTIVE ON ML SYSTEM PERFORMANCE

Christian Kaestner

Carnegie Mellon University

@ FASTPATH 2021

CHRISTIAN KÄSTNER

@p0nk

kaestner@cs.cmu.edu

Associate Professor @ CMU

Interests:

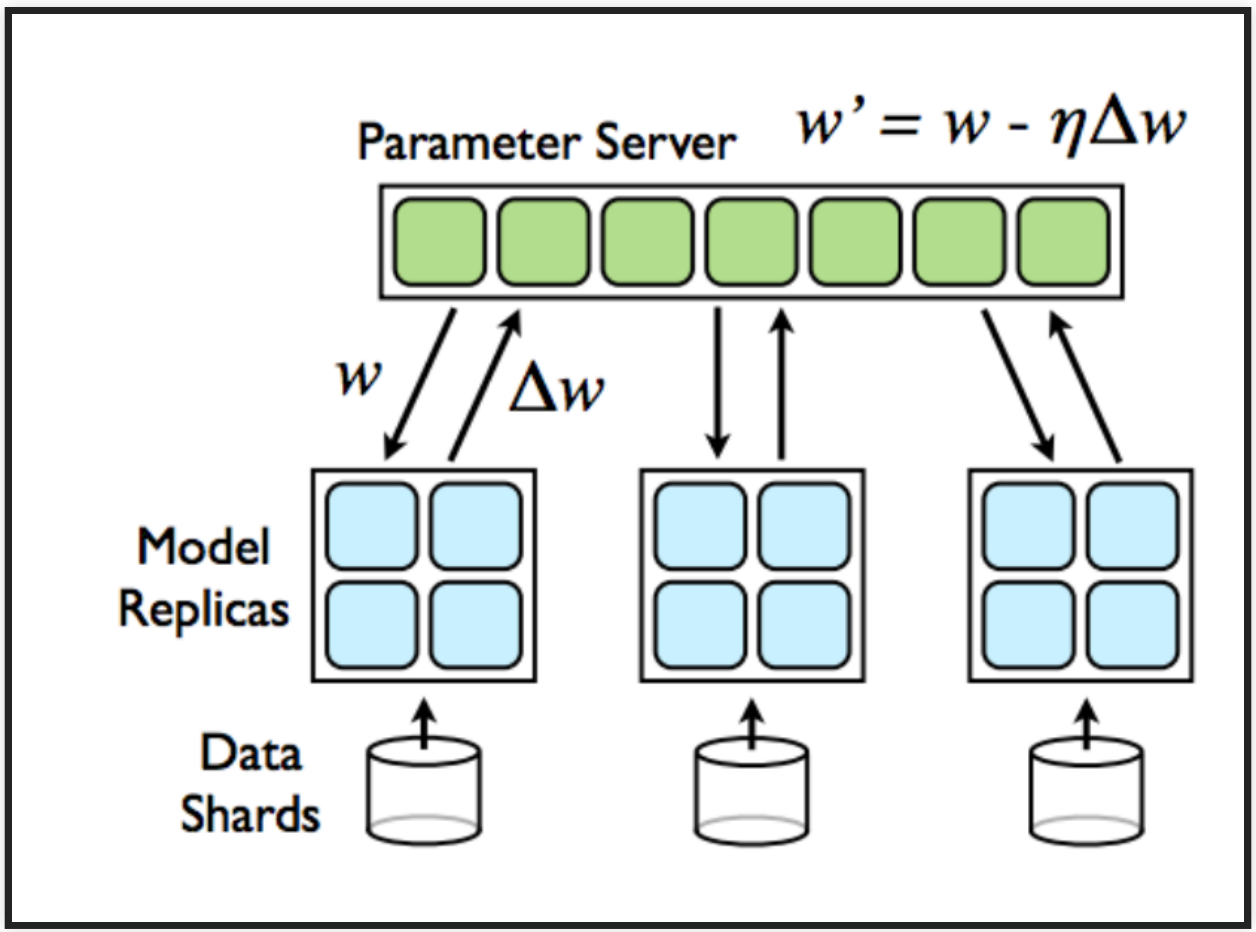
- Software Engineering for ML-Enabled Systems
- Highly-Configurable Systems (incl. performance analysis)
- Sustainability and Stress in Open Source

SOFTWARE ENGINEERING FOR ML-ENABLED SYSTEMS

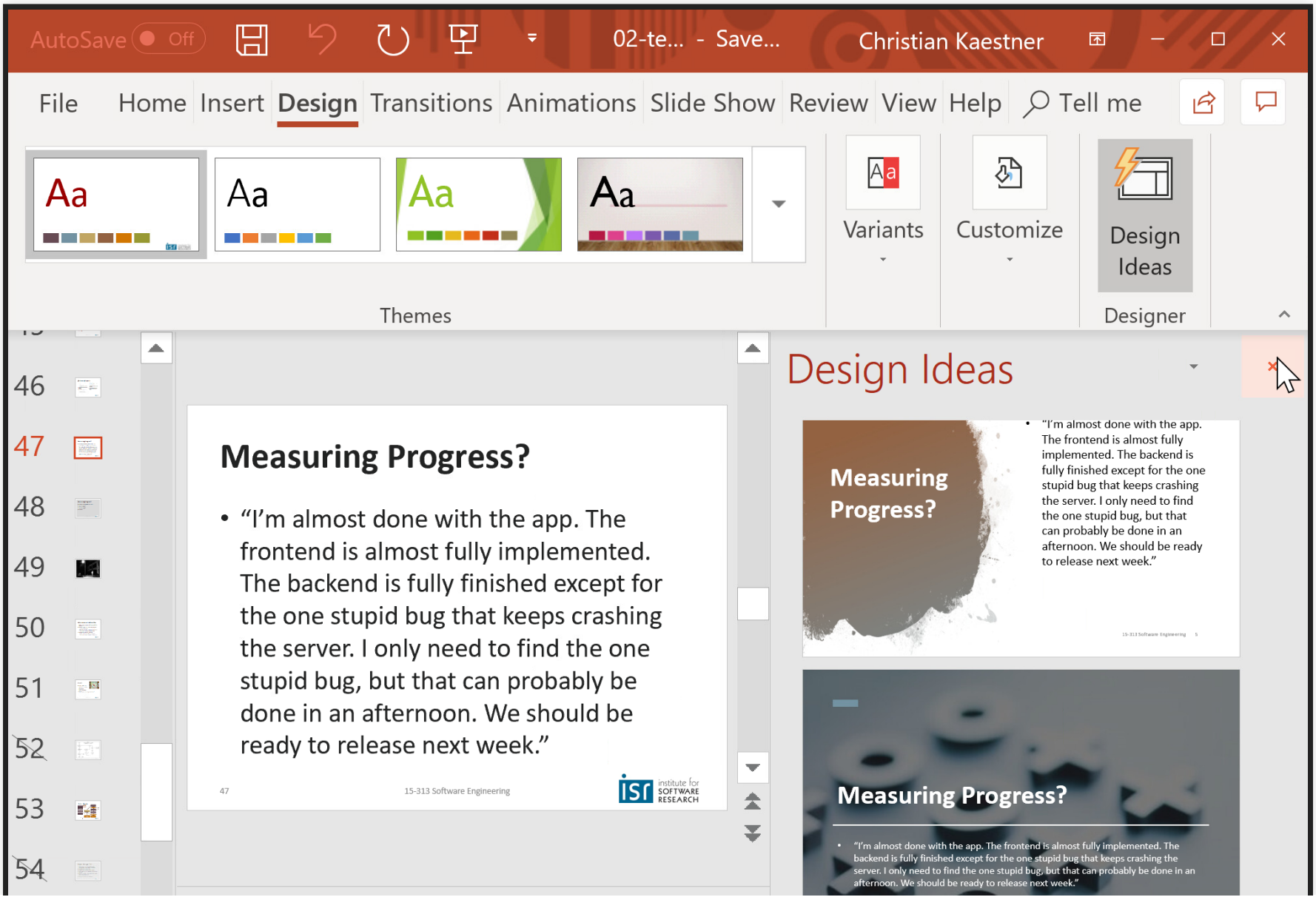
Building, operating, and maintaining software systems with machine-learned components

with interdisciplinary collaborative teams of ***data scientists, software engineers, operators, ...***

SE FOR ML-ENABLED SYSTEMS != DEVELOPING ML FRAMEWORKS



SE FOR ML-ENABLED SYSTEMS



55

56

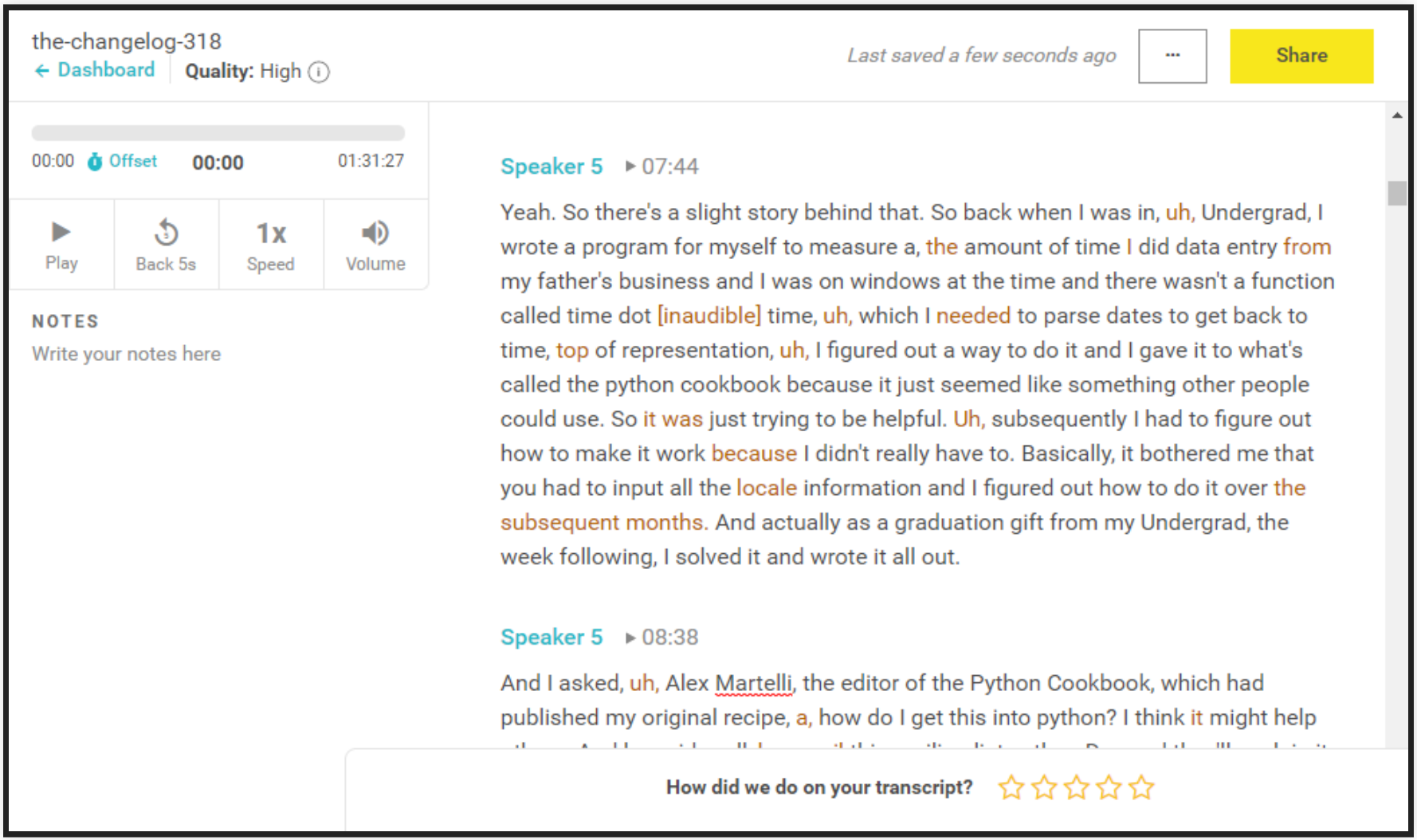
Tap to add notes

Slide 47 of 74

Notes

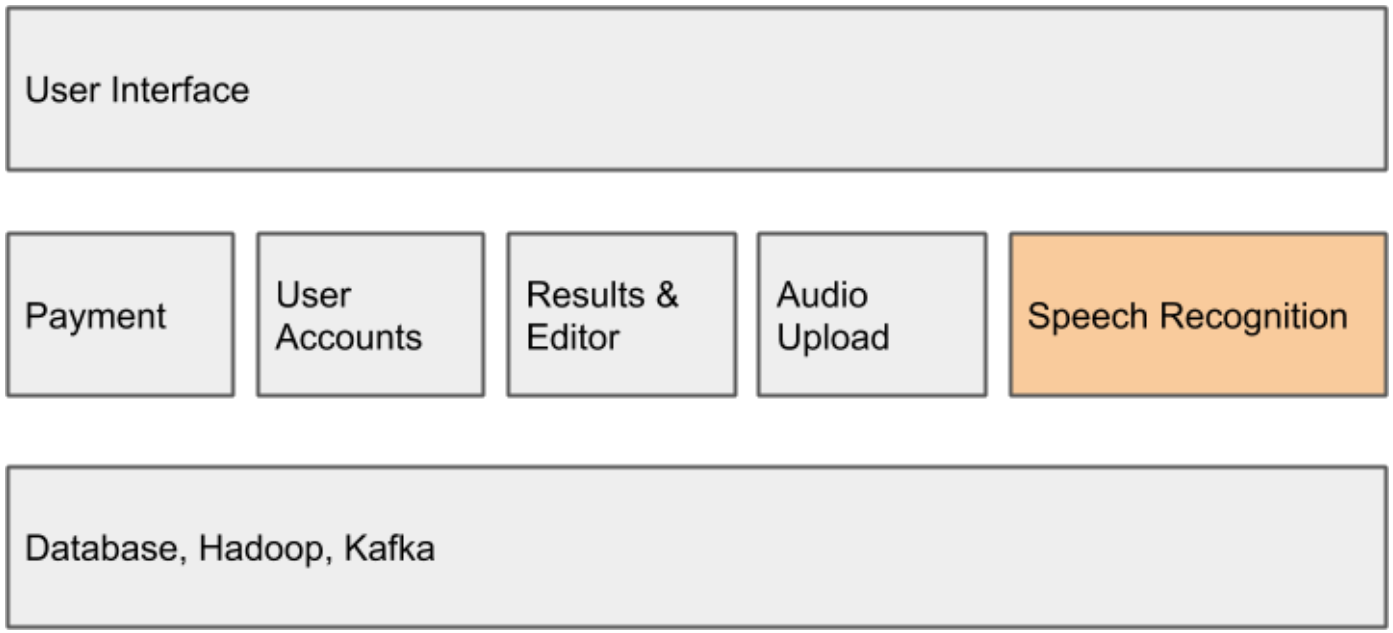
29%

SE FOR ML-ENABLED SYSTEMS



temi.com

SYSTEM = ML + NON-ML COMPONENTS

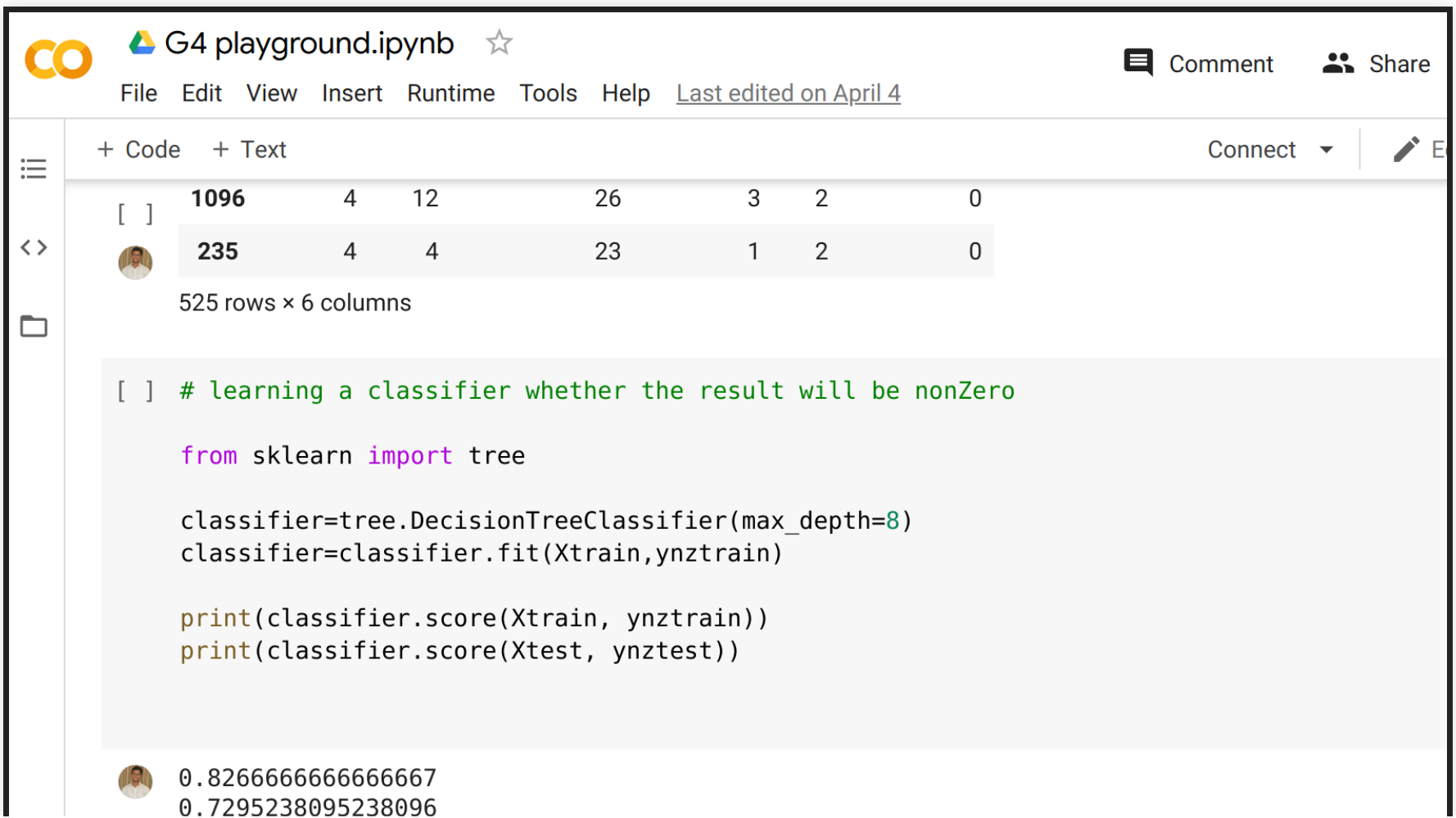


SYSTEM DESIGN MATTERS

MOST ML COURSES

Focus narrowly on modeling techniques or building models

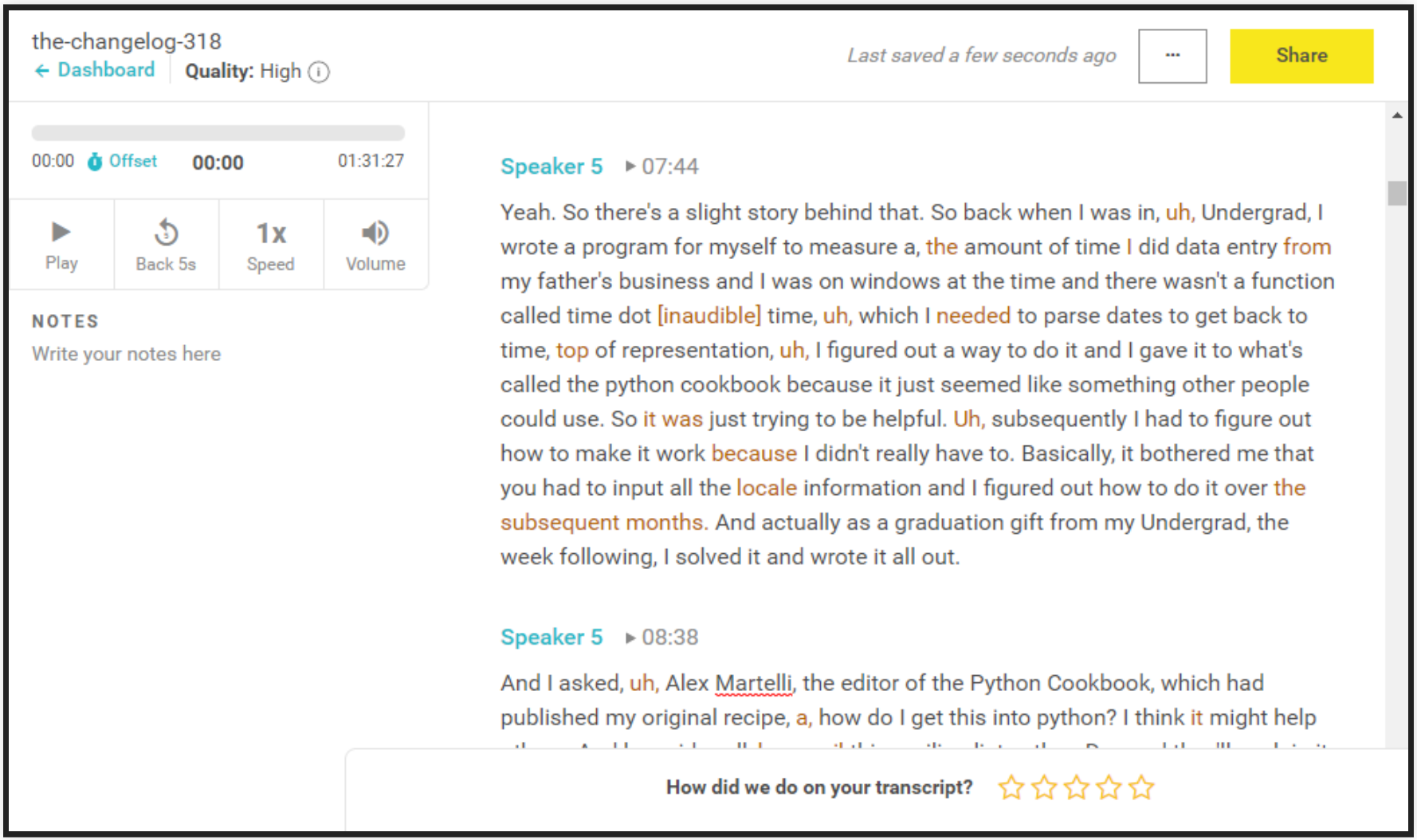
Using notebooks, static datasets, evaluating accuracy



```
[ ] # learning a regression model only on the nonZero data (test is on all data and somewhat

    from sklearn import tree

    predictor=tree.DecisionTreeRegressor(max_depth=8)
    predictor=predictor.fit(XnzTrain,YnzTrain)


    print(predictor.score(XnzTrain, YnzTrain))
    print(predictor.score(Xtest, ytest))
```

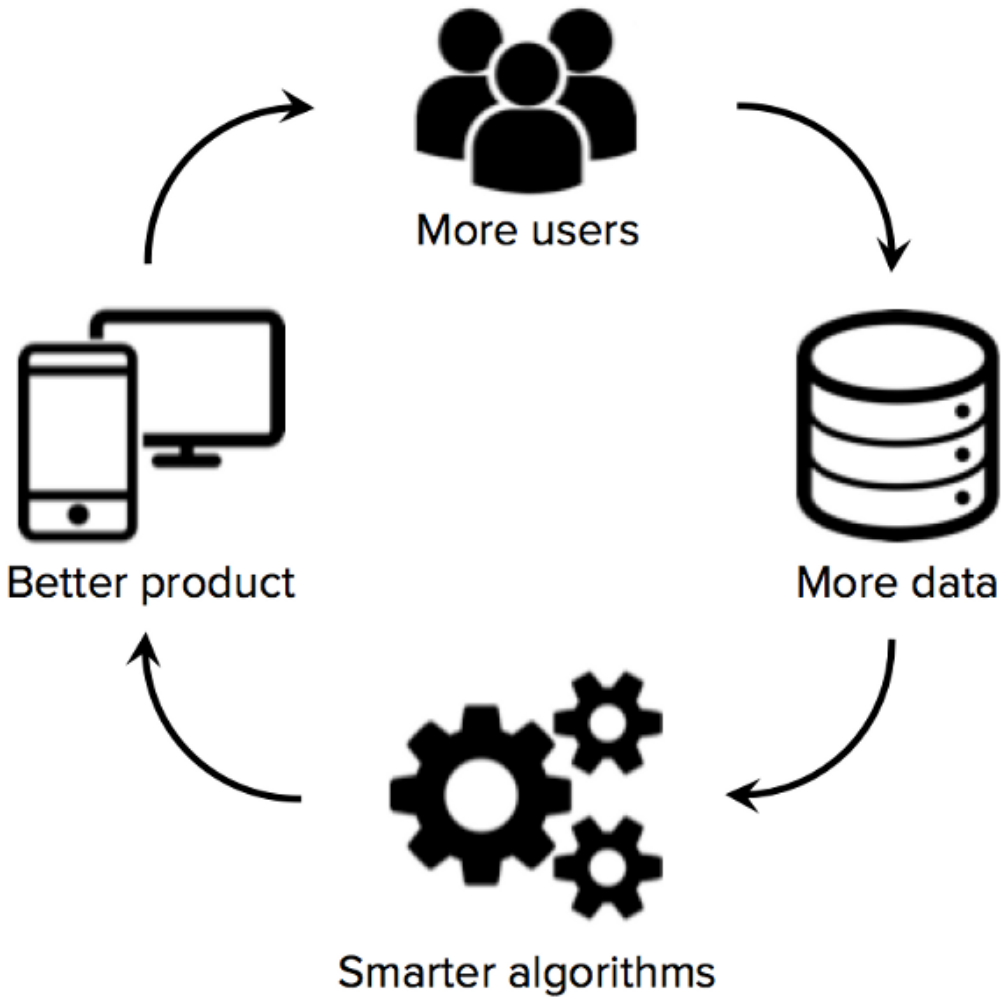
```
0.9376379365613154
-2.437397740412892
```

SE FOR ML-ENABLED SYSTEMS

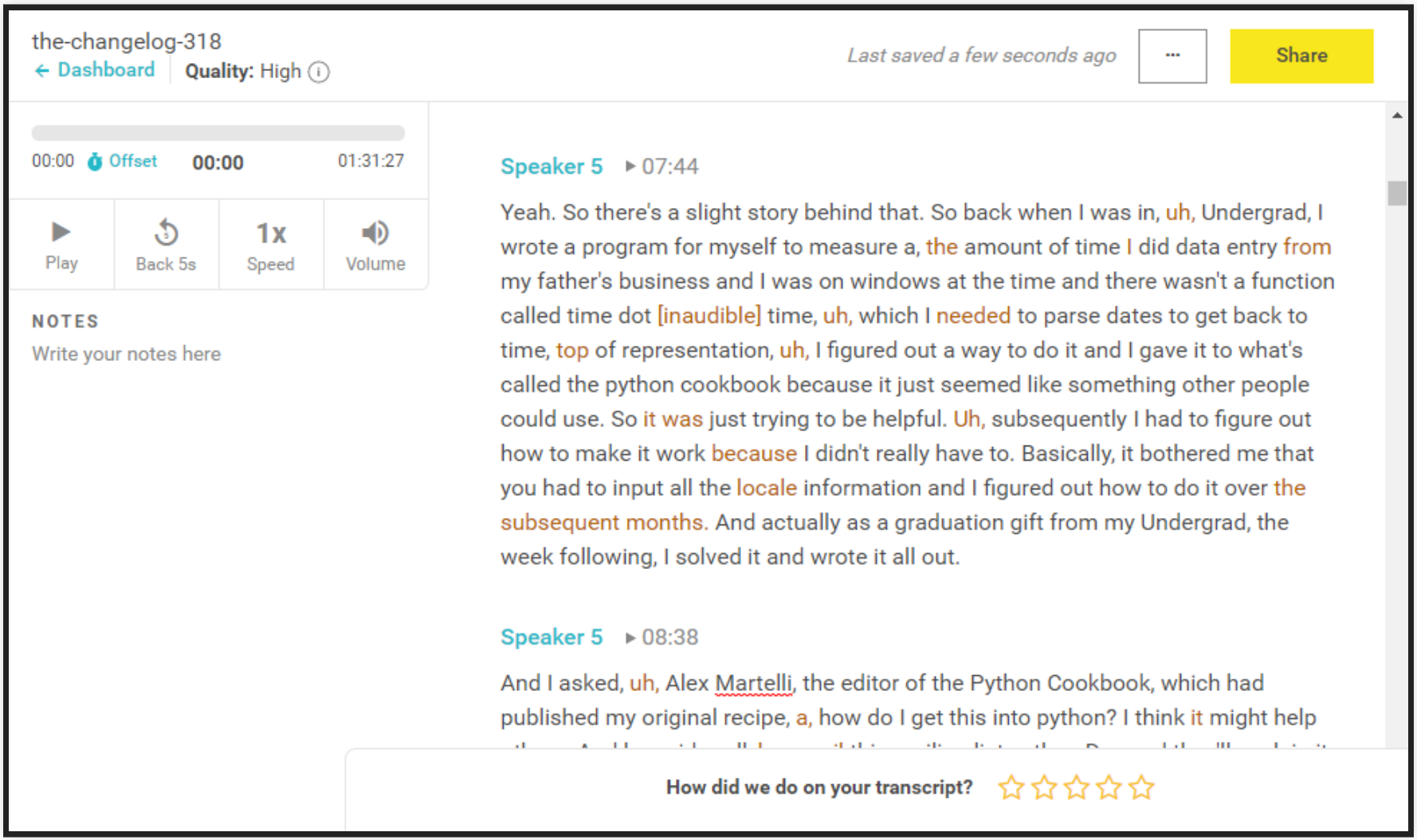


temi.com

THE FLYWHEEL

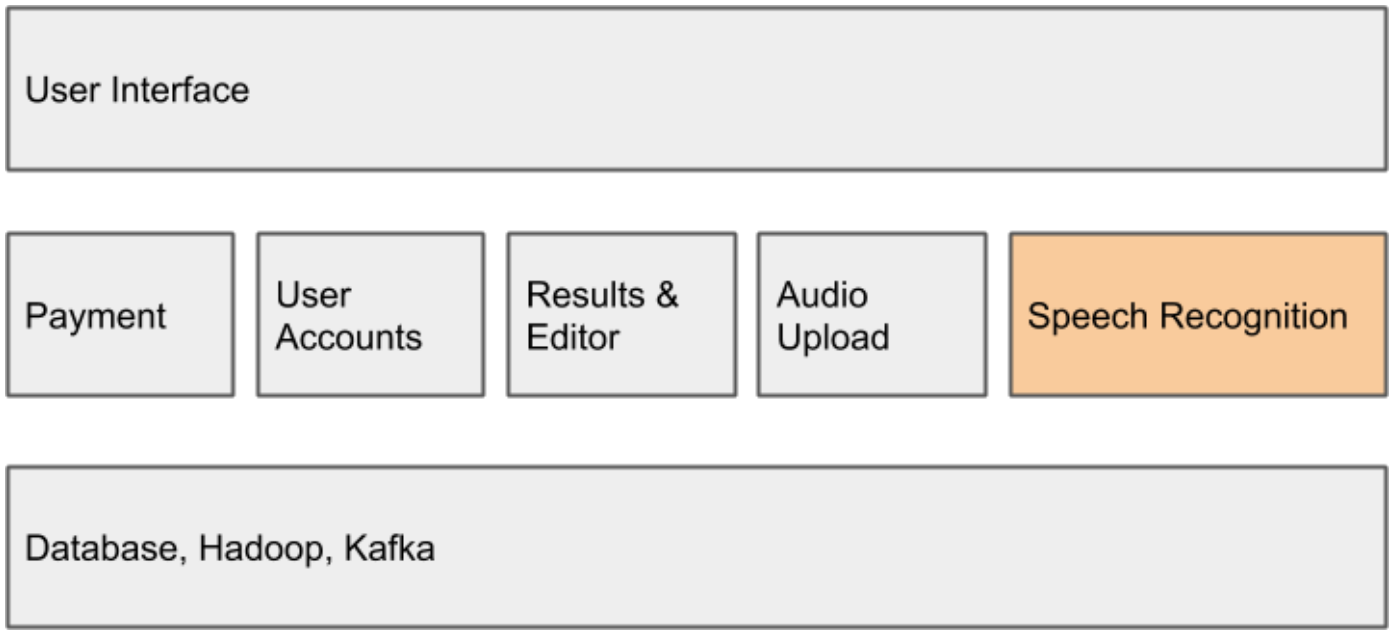


TELEMETRY DESIGN

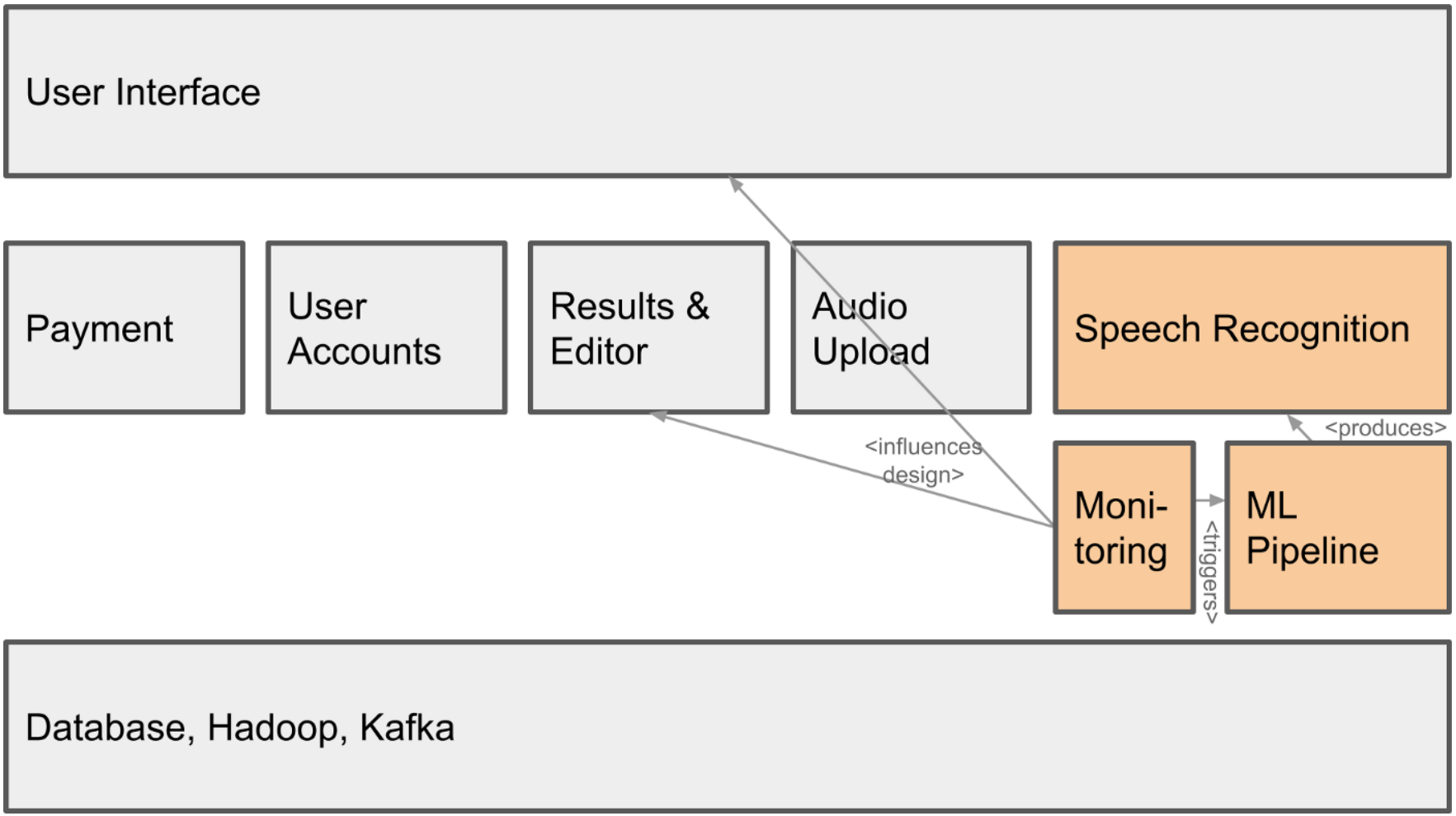


temi.com

ML IS A COMPONENT IN A SYSTEM



SYSTEM DESIGN TO SUPPORT ML



EVERYTHING IS A TRADEOFF

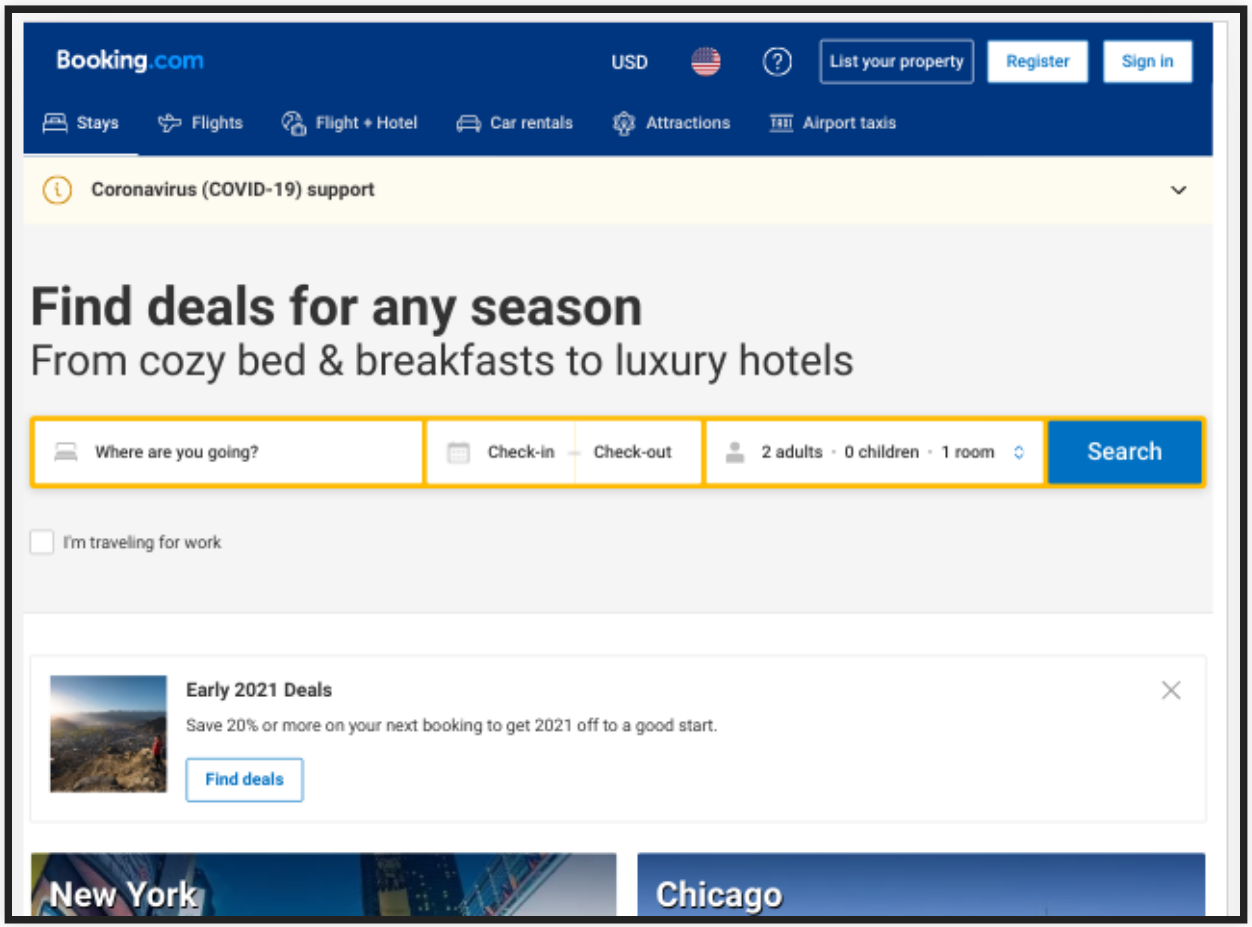
“It Depends.”

MANY QUALITIES OF INTEREST

- Accuracy
- Inference latency, throughput, energy consumption
- Learning time, incremental learning, scalability, resources needed
- Simplicity, maintainability, extensibility,
- Interpretability/explainability, fairness
- Robustness, reproducibility, stability

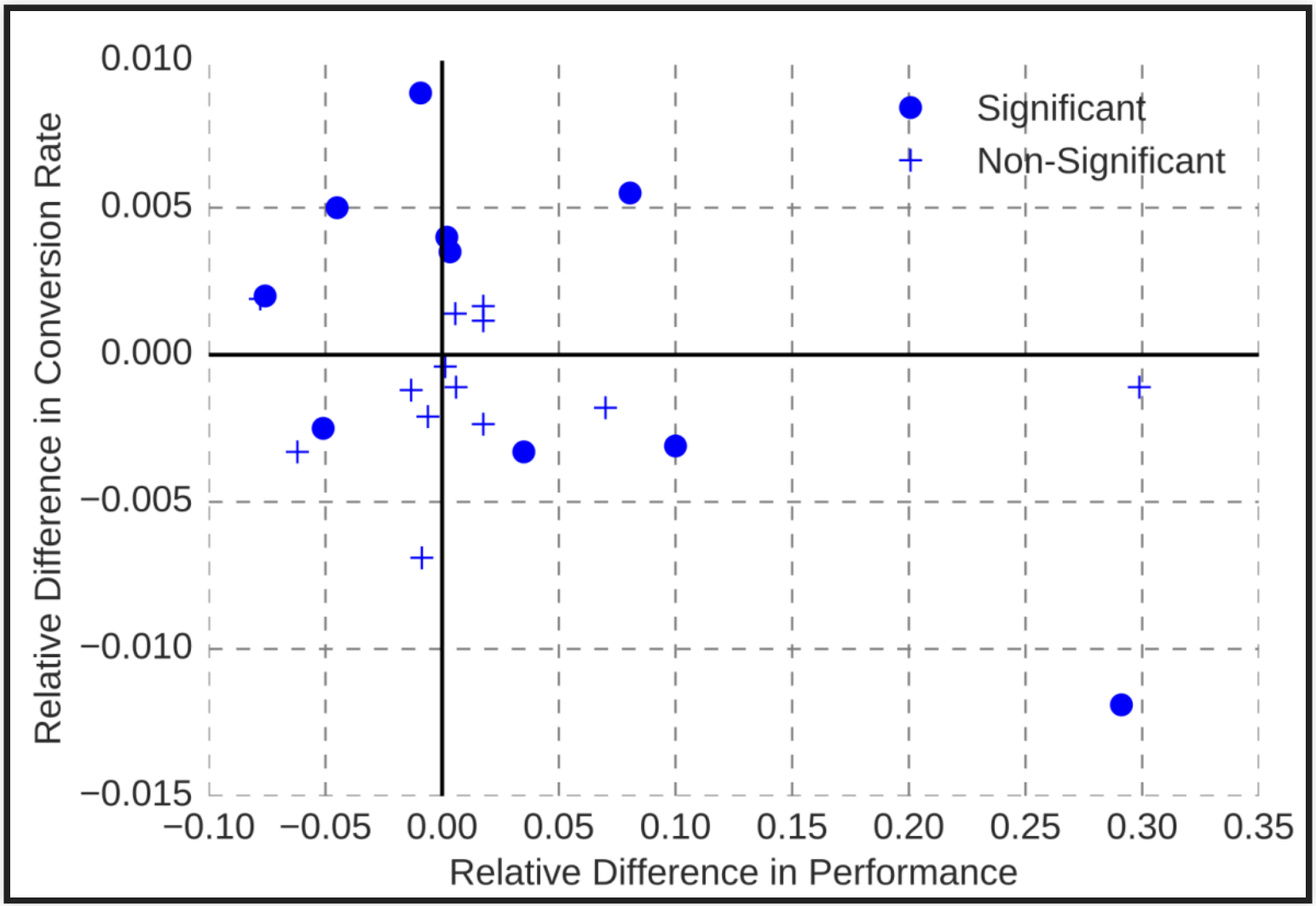
- Usability, trust, accountability
- User satisfaction
- Dealing with mistakes
- Safety, security, privacy
- Development velocity, predictability
- Profit
- ...

MODEL QUALITY VS SYSTEM QUALITY



Bernardi, Lucas, Themistoklis Mavridis, and Pablo Estevez. "150 successful machine learning models: 6 lessons learned at Booking.com." In Proc. International Conference on Knowledge Discovery & Data Mining, 2019.

MODEL QUALITY VS SYSTEM QUALITY



MANY QUALITIES OF INTEREST

- Accuracy
- Inference latency, throughput, energy consumption
- Learning time, incremental learning, scalability, resources needed
- Simplicity, maintainability, extensibility,
- Interpretability/explainability, fairness
- Robustness, reproducibility, stability

- Usability, trust, accountability
- User satisfaction
- Dealing with mistakes
- Safety, security, privacy
- Development velocity, predictability
- Profit
- ...

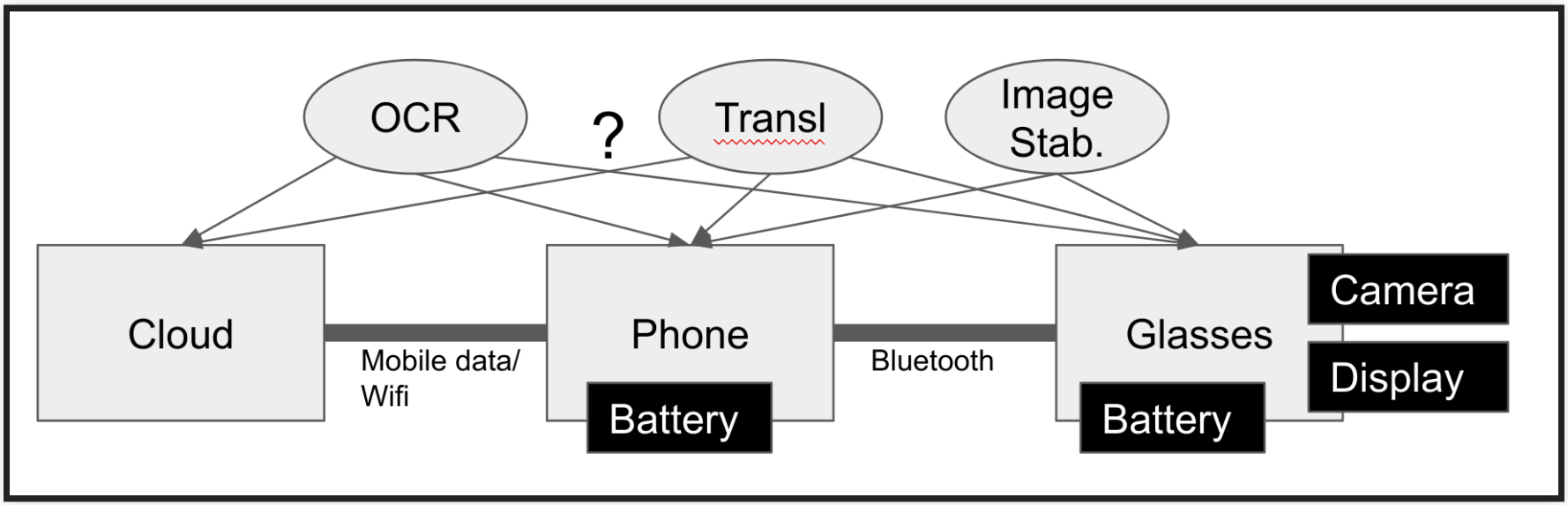
EXAMPLE: AR INSTANT TRANSLATION

EXAMPLE: AR INSTANT TRANSLATION

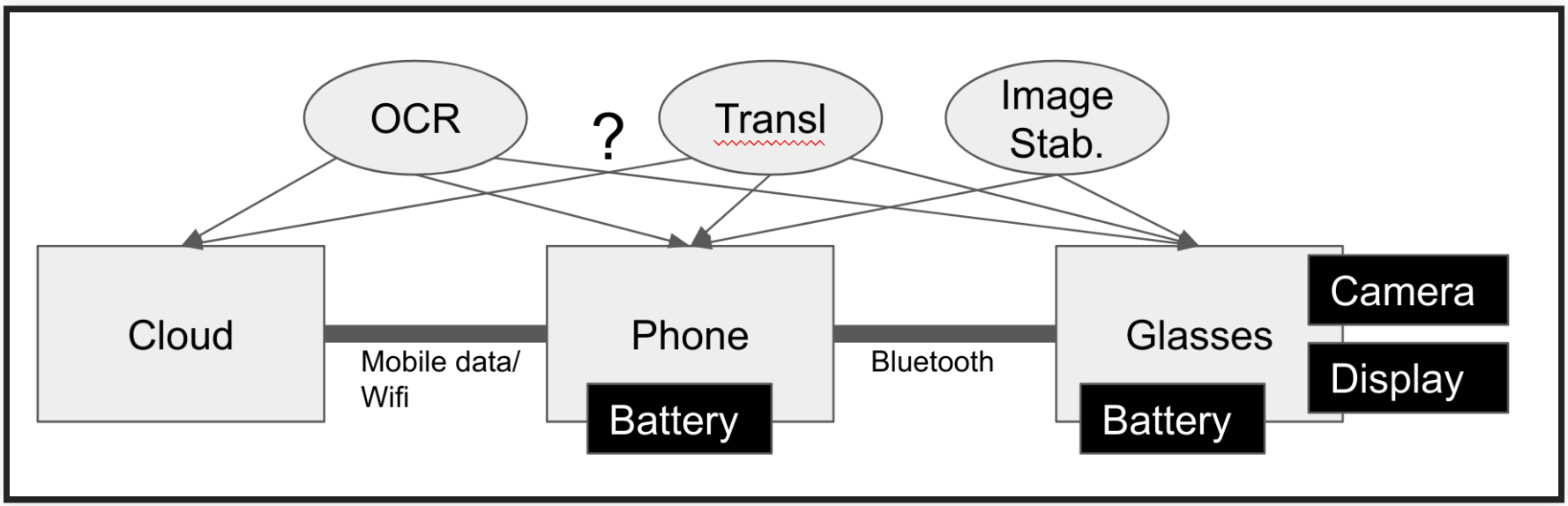
EXAMPLE: AR INSTANT TRANSLATION

EXAMPLE: AR INSTANT TRANSLATION

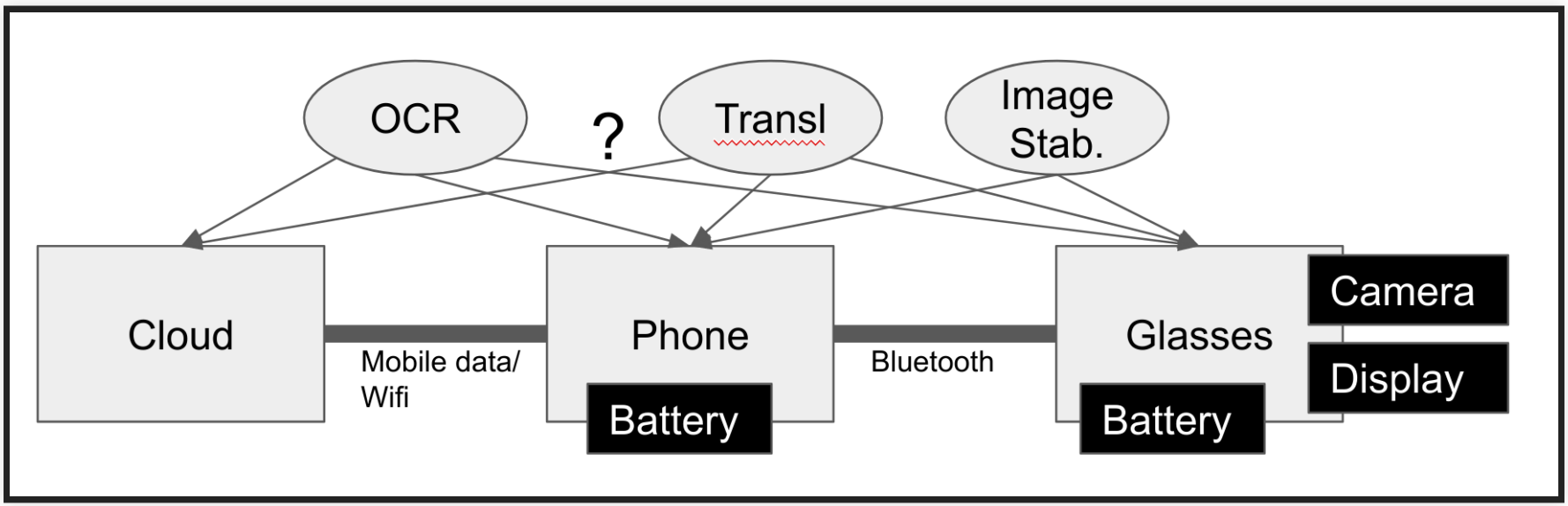
EXAMPLE: AR INSTANT TRANSLATION



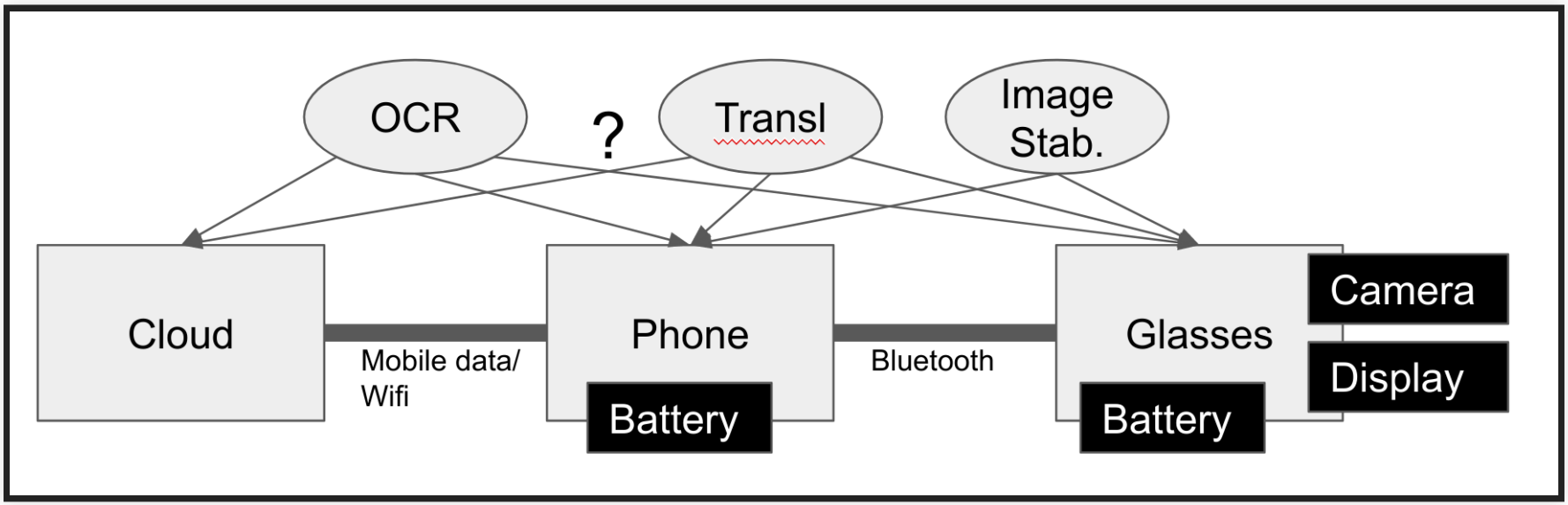
TRADEOFF: ACCURACY VS LATENCY + ENERGY + MODEL SIZE



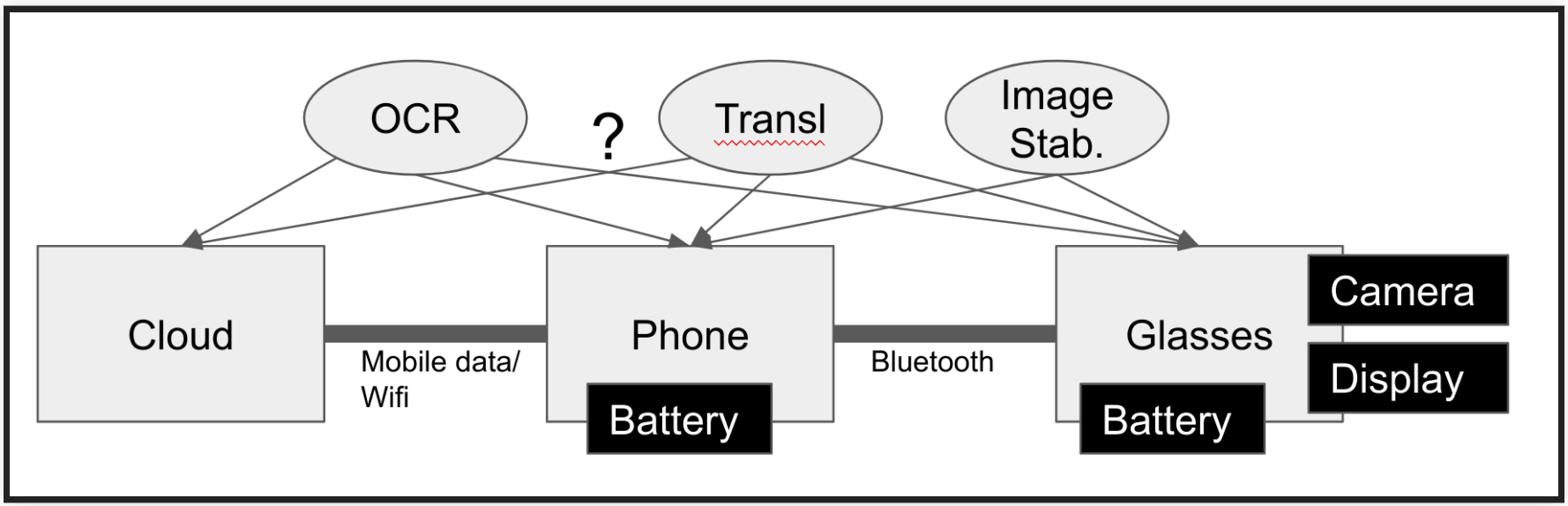
TRADEOFF: LATENCY VS ENERGY CONSUMPTION VS BANDWIDTH



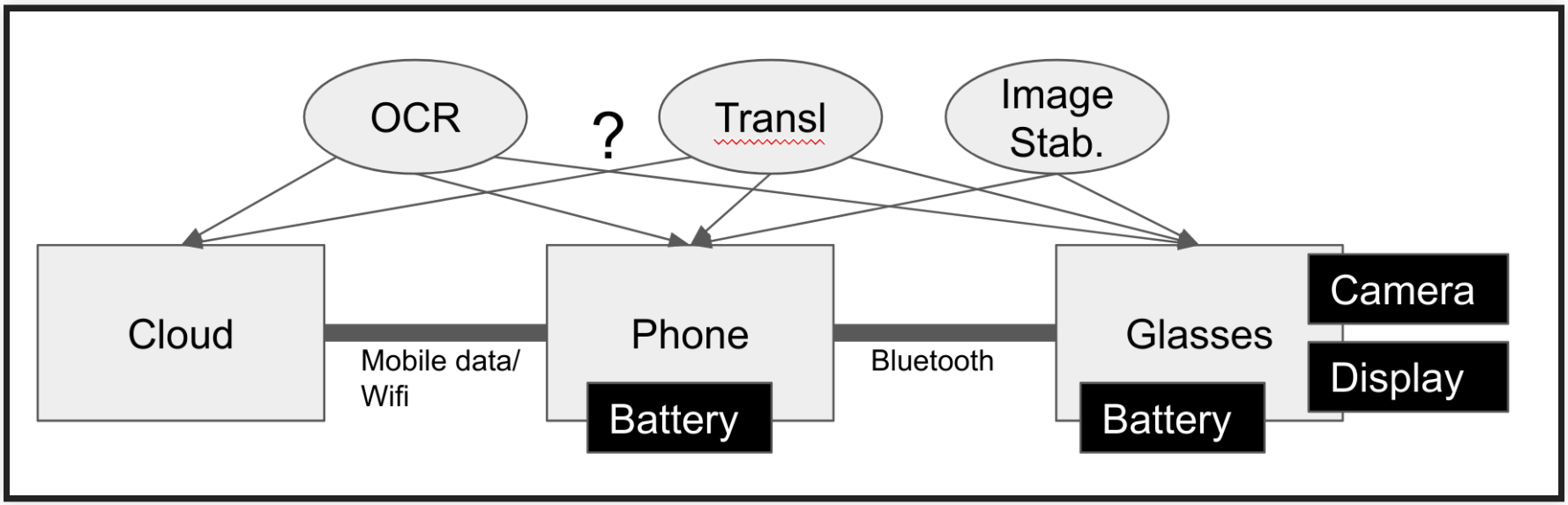
TRADEOFF: PRIVACY VS TELEMETRY



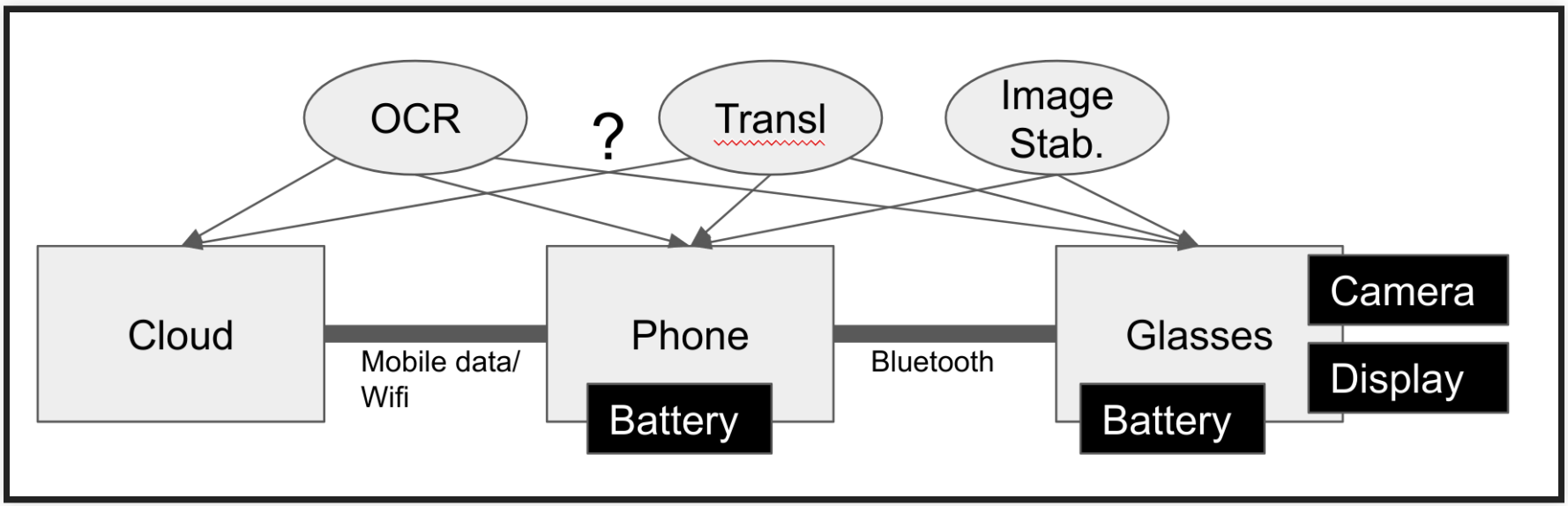
TRADEOFF: TELEMETRY BENEFITS VS TELEMETRY COSTS



TRADEOFF: UPDATE LATENCY VS OFFLINE USE

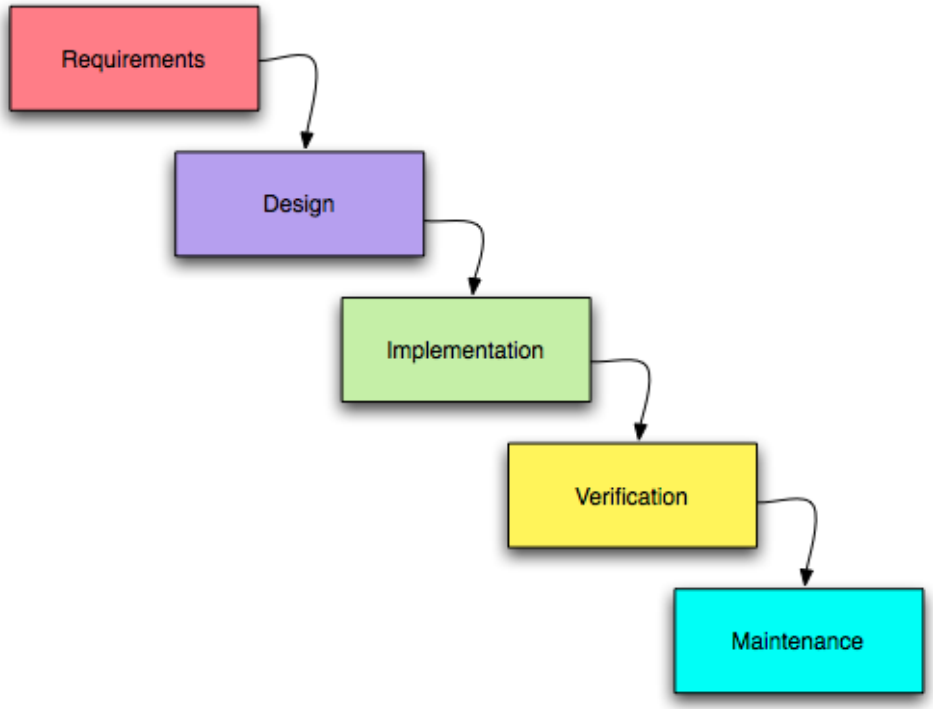


TRADEOFF: TRAINING COST VS UPDATE FREQUENCY



“It Depends.”

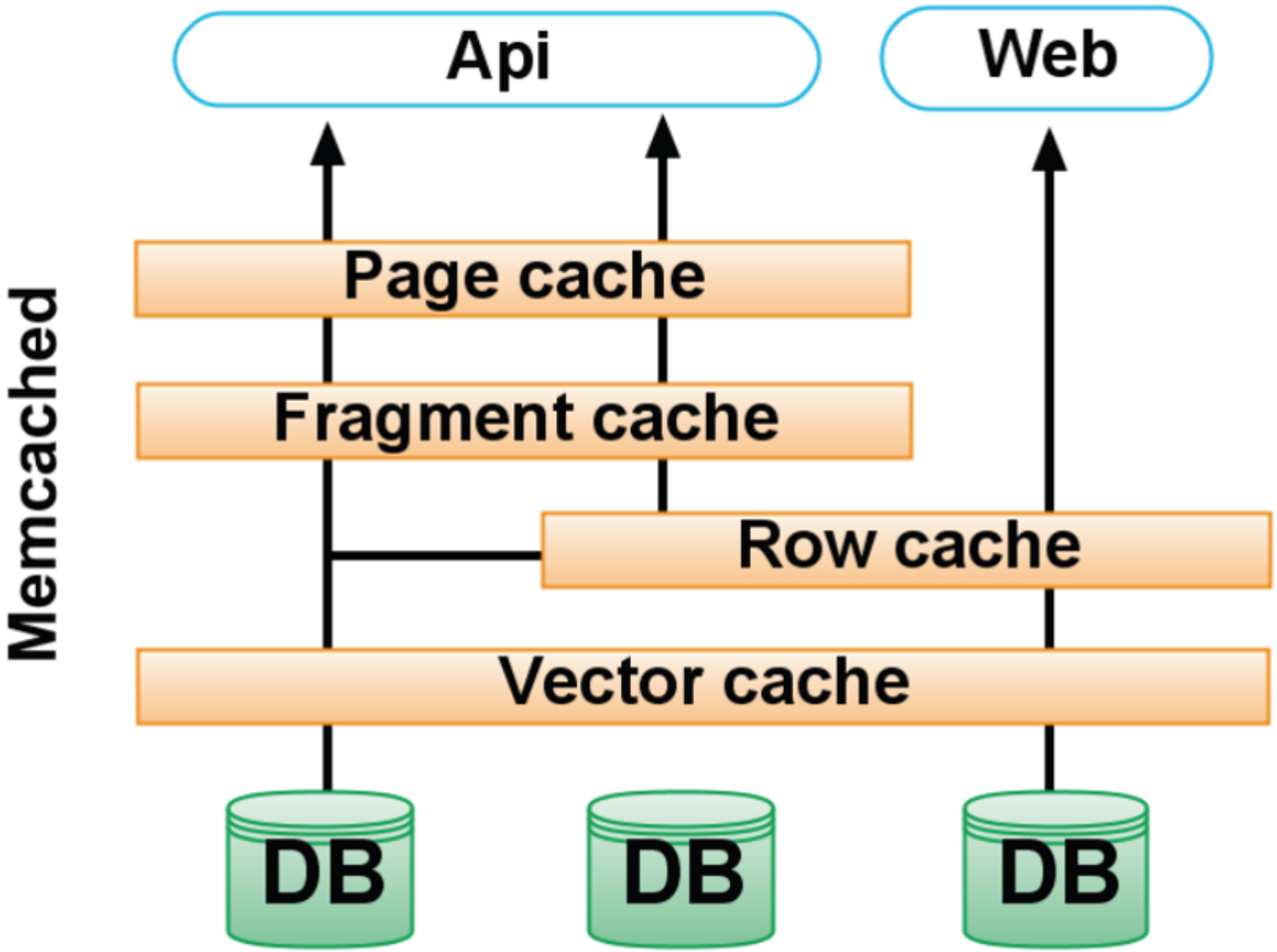
THINK LIKE A SOFTWARE ARCHITECT


Requirements
Design
Implementation
Verification
Maintenance

EXCURSION: TWITTER

Raffi. New Tweets per second record, and how! Twitter Blog, 2013

TWITTER - CACHING ARCHITECTURE



Speaker notes

- Running one of the world’s largest Ruby on Rails installations
- 200 engineers
- Monolithic: managing raw database, memcache, rendering the site, and * presenting the public APIs in one codebase
- Increasingly difficult to understand system; organizationally challenging to manage and parallelize engineering teams
- Reached the limit of throughput on our storage systems (MySQL); read and write hot spots throughout our databases
- Throwing machines at the problem; low throughput per machine (CPU + RAM limit, network not saturated)
- Optimization corner: trading off code readability vs performance

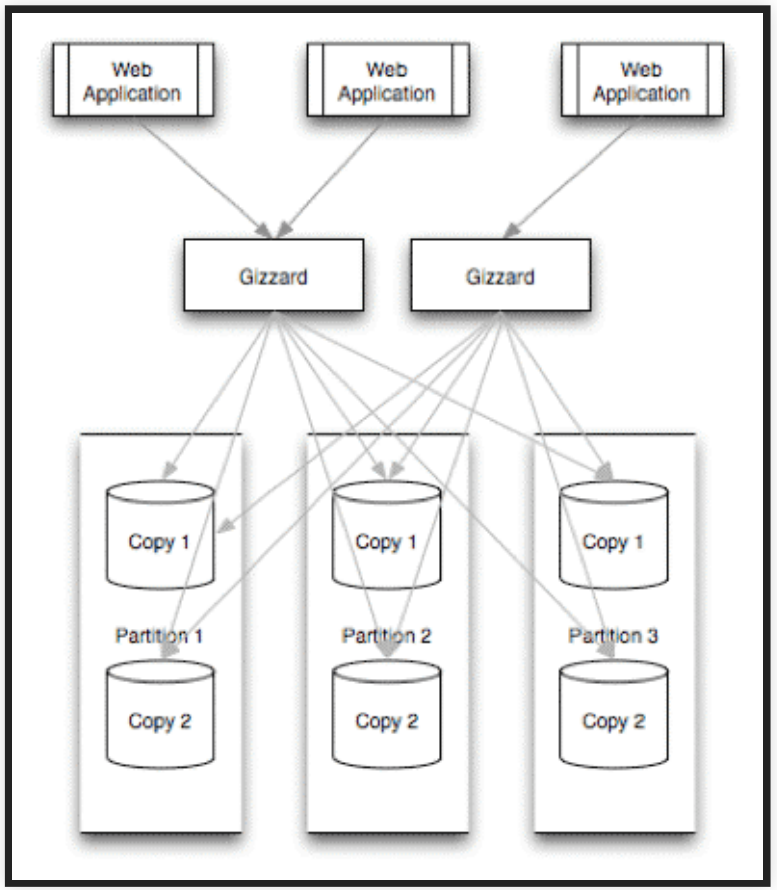
TWITTER'S REDESIGN GOALS

- Performance
  - Improve median latency; lower outliers
  - Reduce number of machines 10x
- Reliability
  - Isolate failures
- Maintainability
  - "We wanted cleaner boundaries with “related” logic being in one place": encapsulation and modularity at the systems level (rather than at the class, module, or package level)
- Modifiability
  - Quicker release of new features: "run small and empowered engineering teams that could make local decisions and ship user-facing changes, independent of other teams"

Raffi. New Tweets per second record, and how! Twitter Blog, 2013

TWITTER: REDESIGN DECISIONS

- Ruby on Rails -> JVM/Scala
- Monolith -> Microservices
- RPC framework with monitoring, connection pooling, failover strategies, loadbalancing, ... built in
- New storage solution, temporal clustering, "roughly sortable ids"
- Data driven decision making

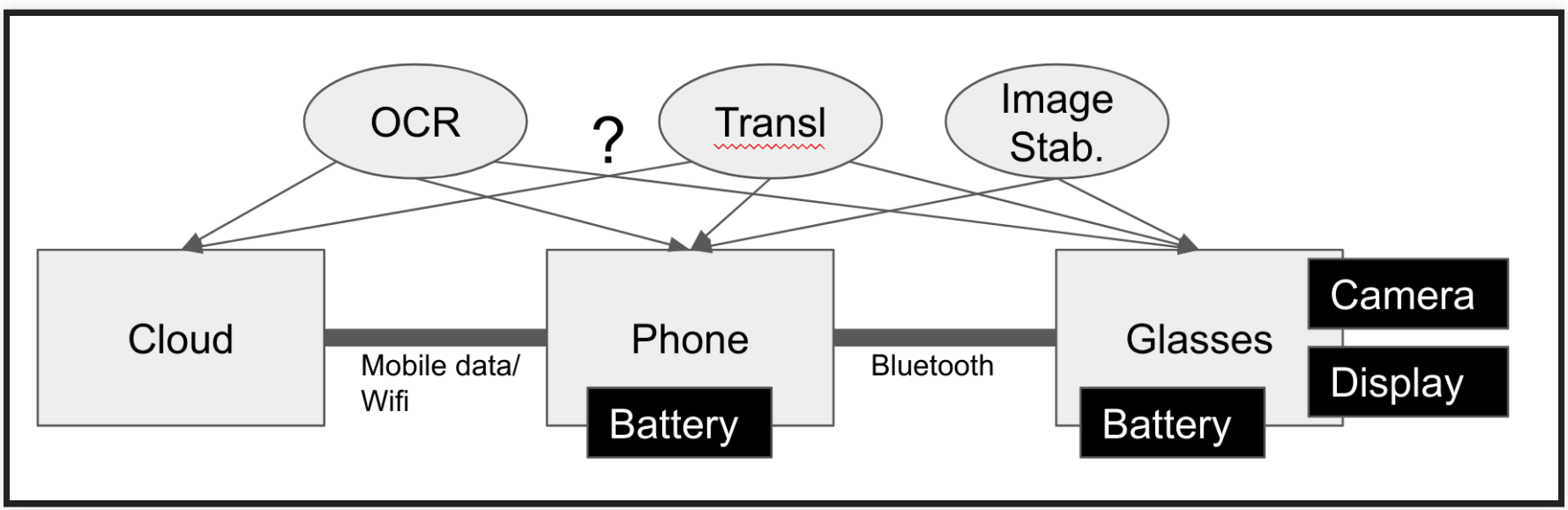


TWITTER CASE STUDY: KEY INSIGHTS

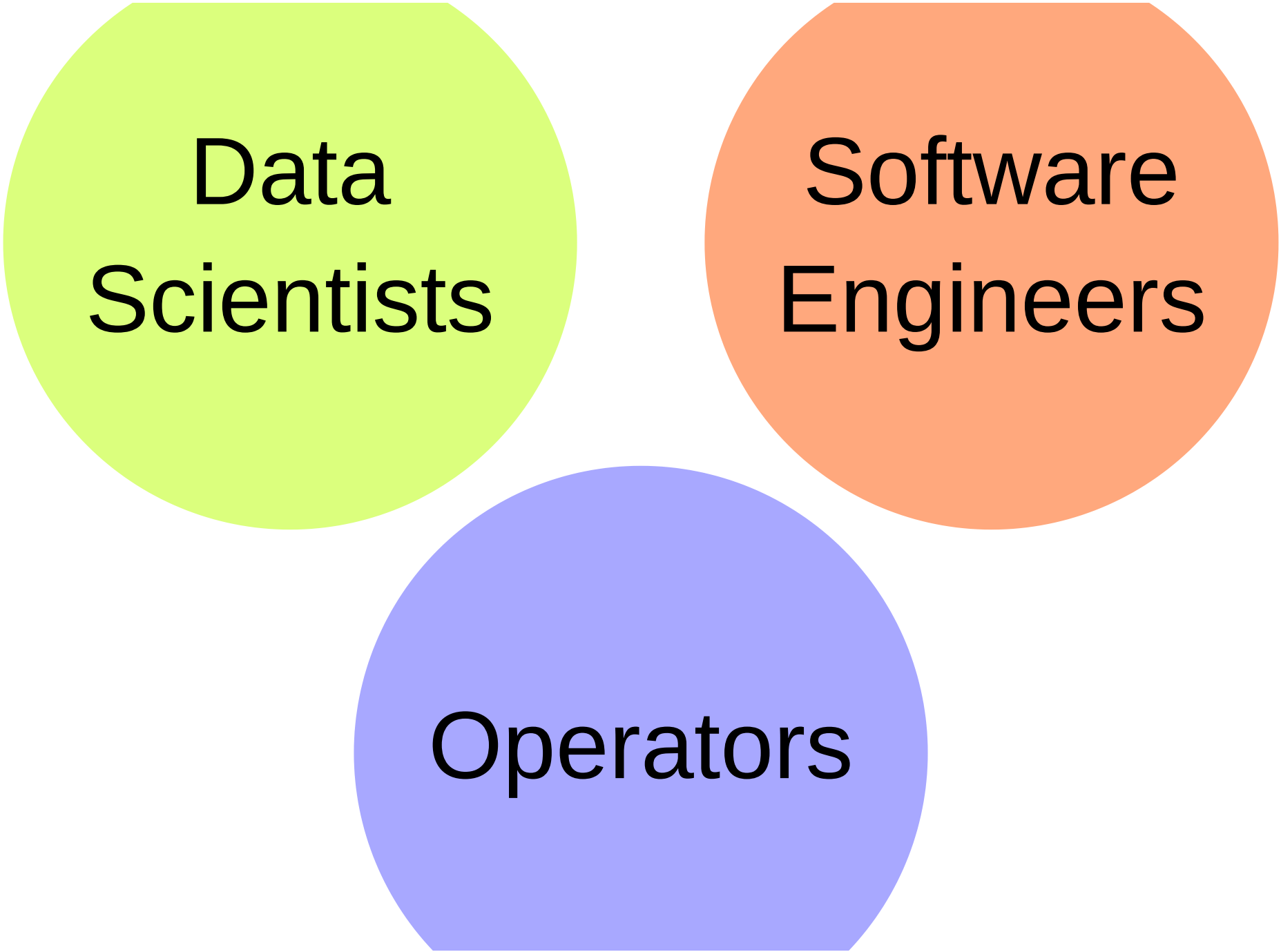
- Architectural decisions affect entire systems, not only individual modules
- Abstract, different abstractions for different scenarios
- Reason about quality attributes early
- Make architectural decisions explicit

ARCHITECTURAL PLANNING

- Identify and prioritize relevant qualities
- Identify system structure and relevant interactions
- Understand constraints and tradeoffs
- Conduct research into requirements and constraints
- Explore alternatives
- Set obligations for components



TEAMS AND PROCESS

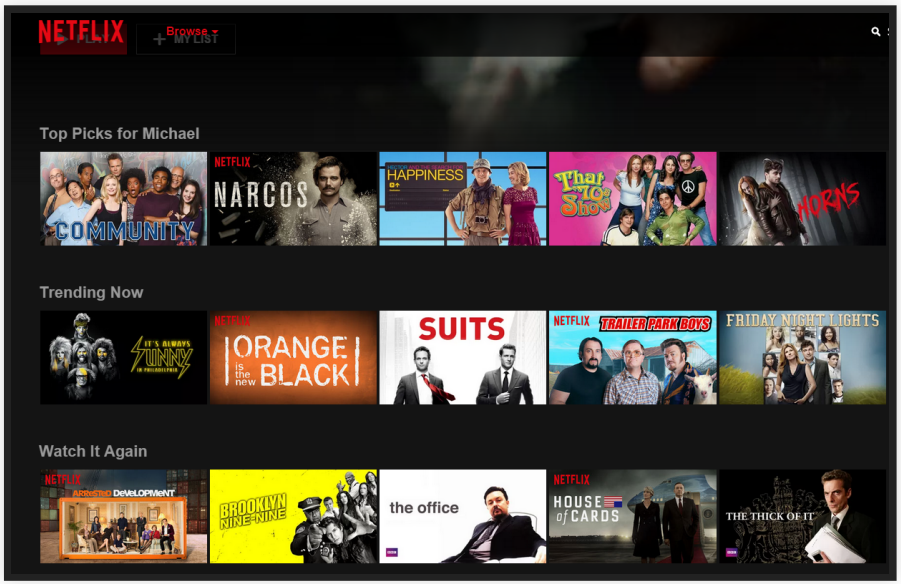

Data Scientists
Software Engineers
Operators

DATA SCIENTIST

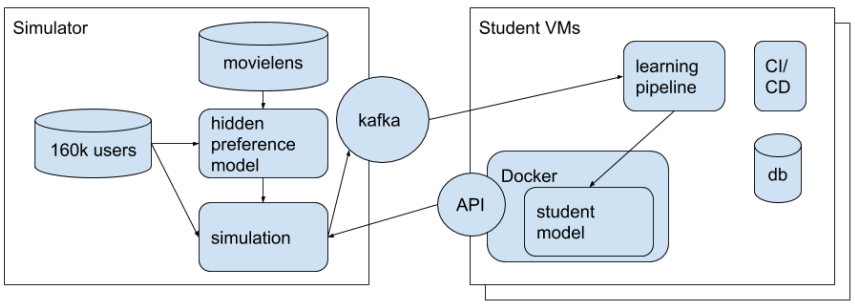
- Often fixed dataset for training and evaluation (e.g., PBS interviews)
- Focused on accuracy
- Prototyping, often Jupyter notebooks or similar
- Expert in modeling techniques and feature engineering
- Model size, updateability, implementation stability typically does not matter
- Starting to worry about fairness, robustness, ...

SOFTWARE ENGINEER

- Builds a product
- Concerned about cost, performance, stability, release time
- Identify quality through customer satisfaction
- Must scale solution, handle large amounts of data
- Plan for mistakes and safeguards
- Maintain, evolve, and extend the product over long periods
- Consider requirements for security, safety, fairness

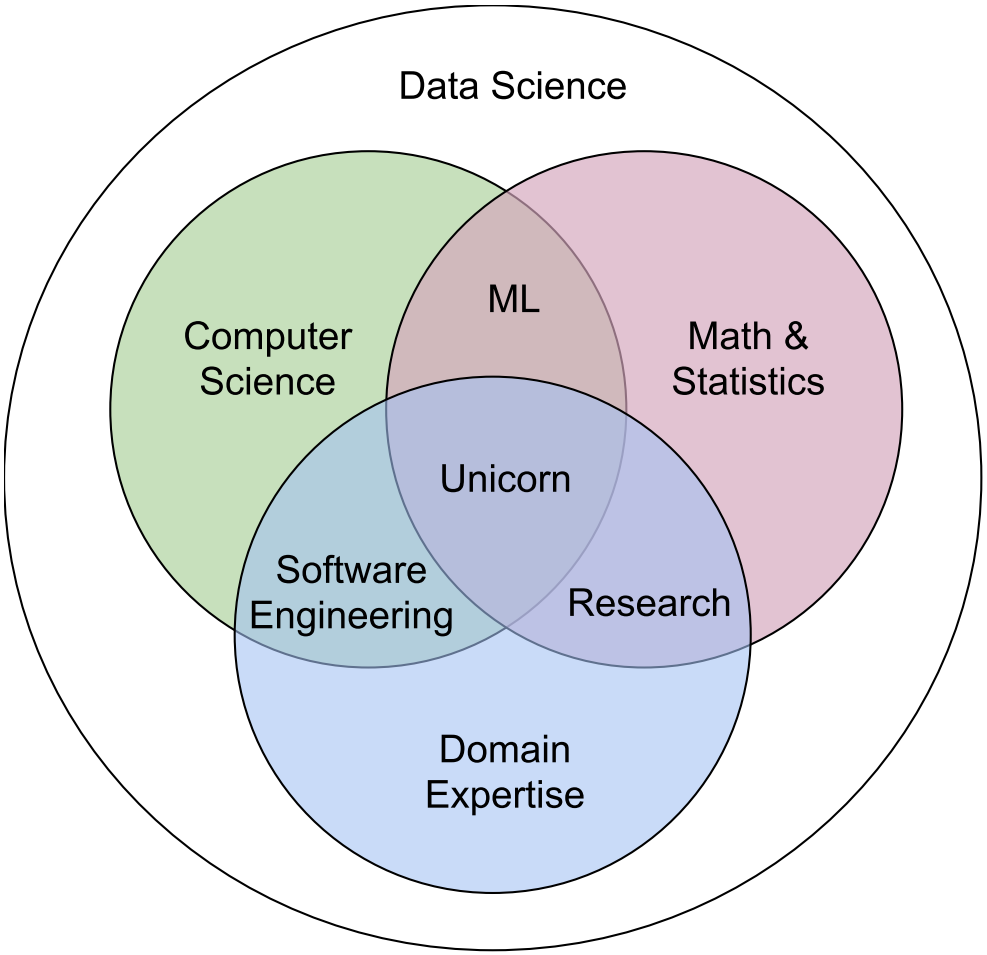


CMU 17-445 Class Project



DATA SCIENCE PRACTICES BY SOFTWARE ENGINEERS

- Many software engineers get involved in data science without explicit training
- Copying from public examples, little reading of documentation
- Lack of data visualization/exploration/understanding, no focus on data quality
- Strong preference for code editors, non-GUI tools
- Try improving model by adding more data or using deep learning, rarely feature engineering or debugging
- Lack of awareness about overfitting/bias problems, single focus on accuracy, no monitoring



By Steven Geringer, via Ryan Orban. Bridging the Gap Between Data Science & Engineer: Building High-Performance Teams. 2016

T-SHAPED PEOPLE

Broad-range generalist + Deep expertise

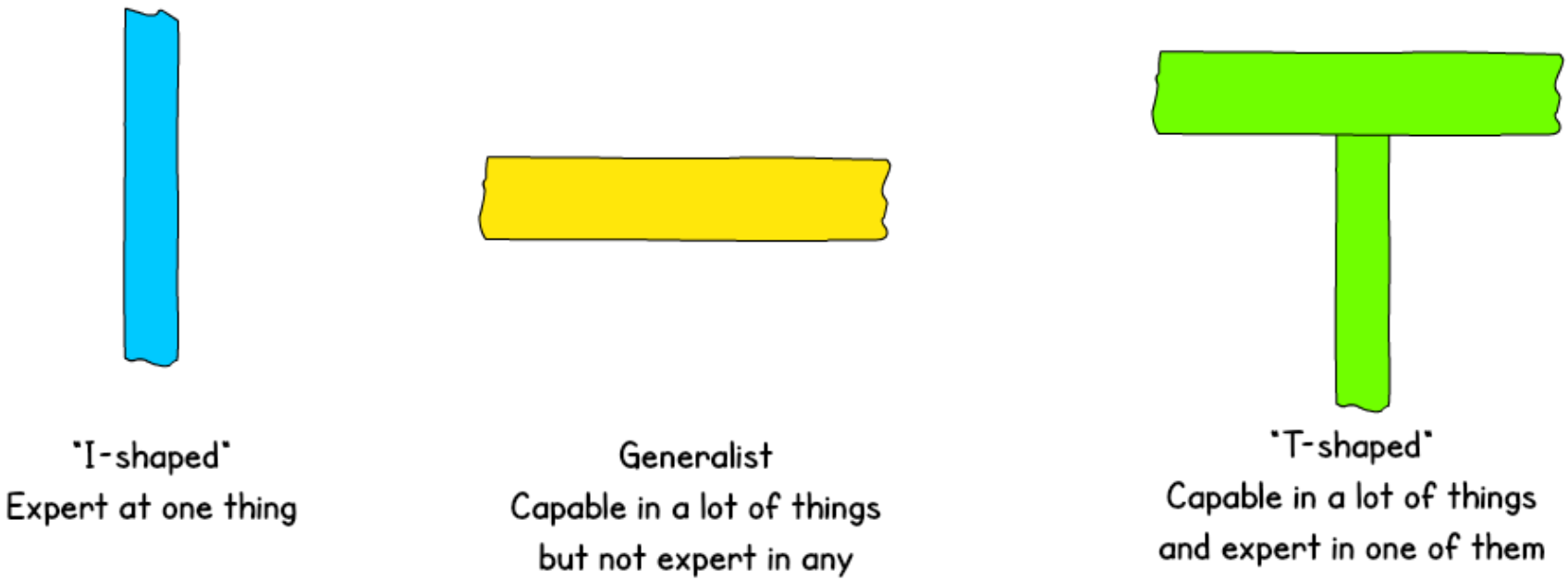
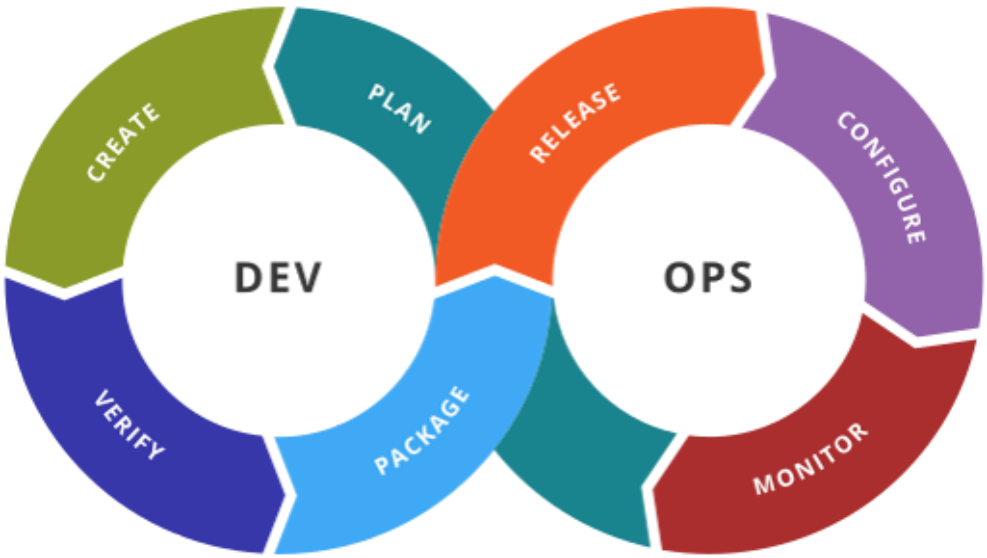


Figure: Jason Yip. Why T-shaped people?. 2018

DEVOPS AS INSPIRATION



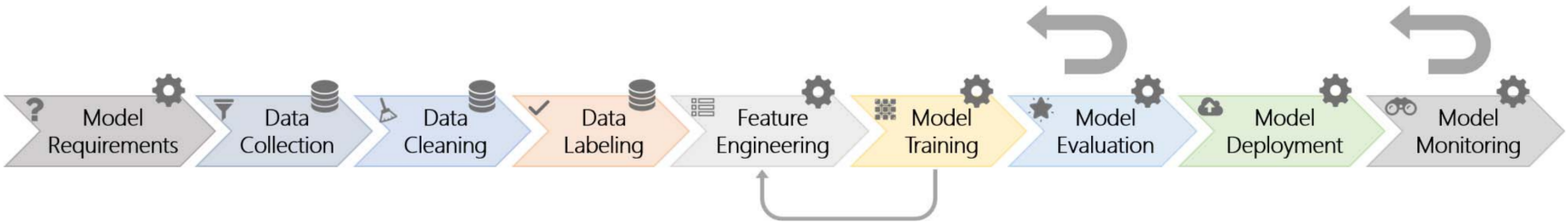
Distinct roles and expertise, but joint responsibilities, joint tooling

Interdisciplinary teams, split expertise, but joint responsibilities

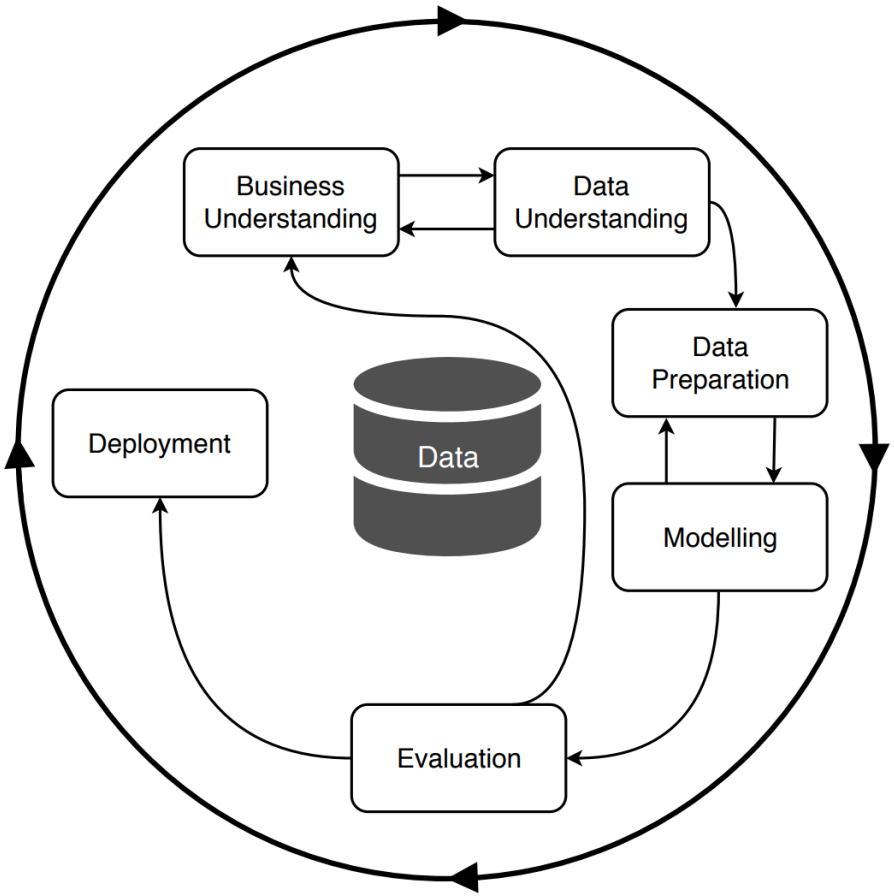
Joint vocabulary and tools

Foster system thinking

PROCESS MODELS

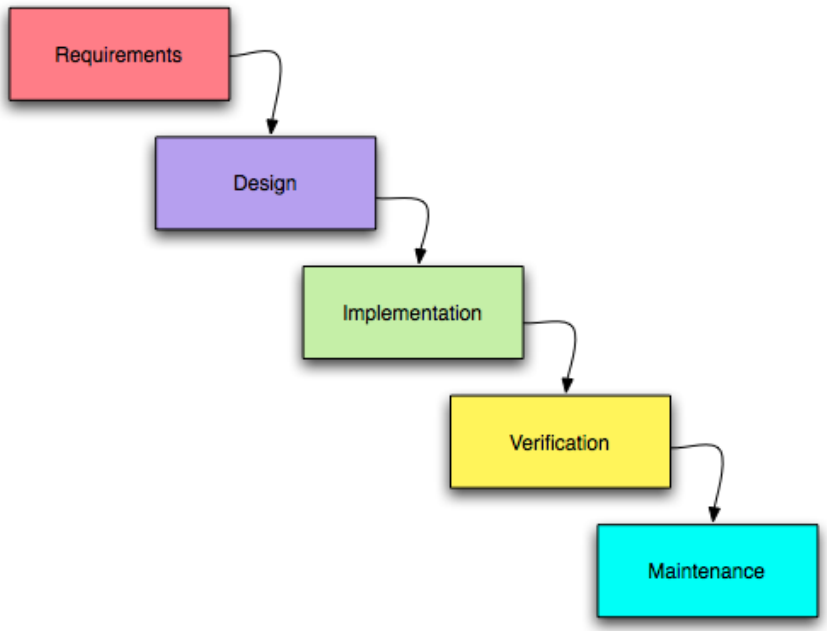


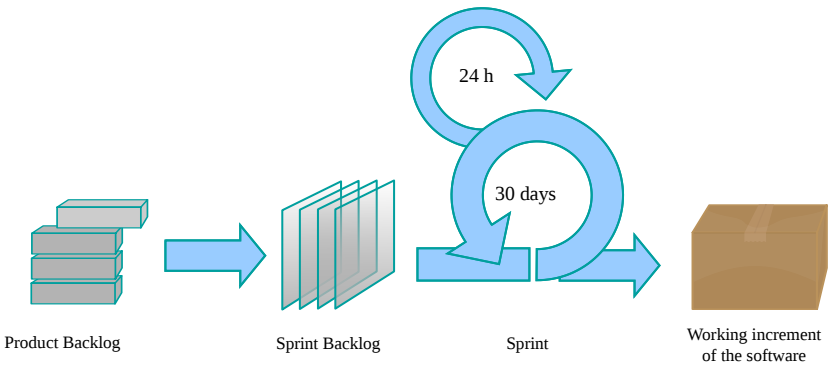
PROCESS MODELS



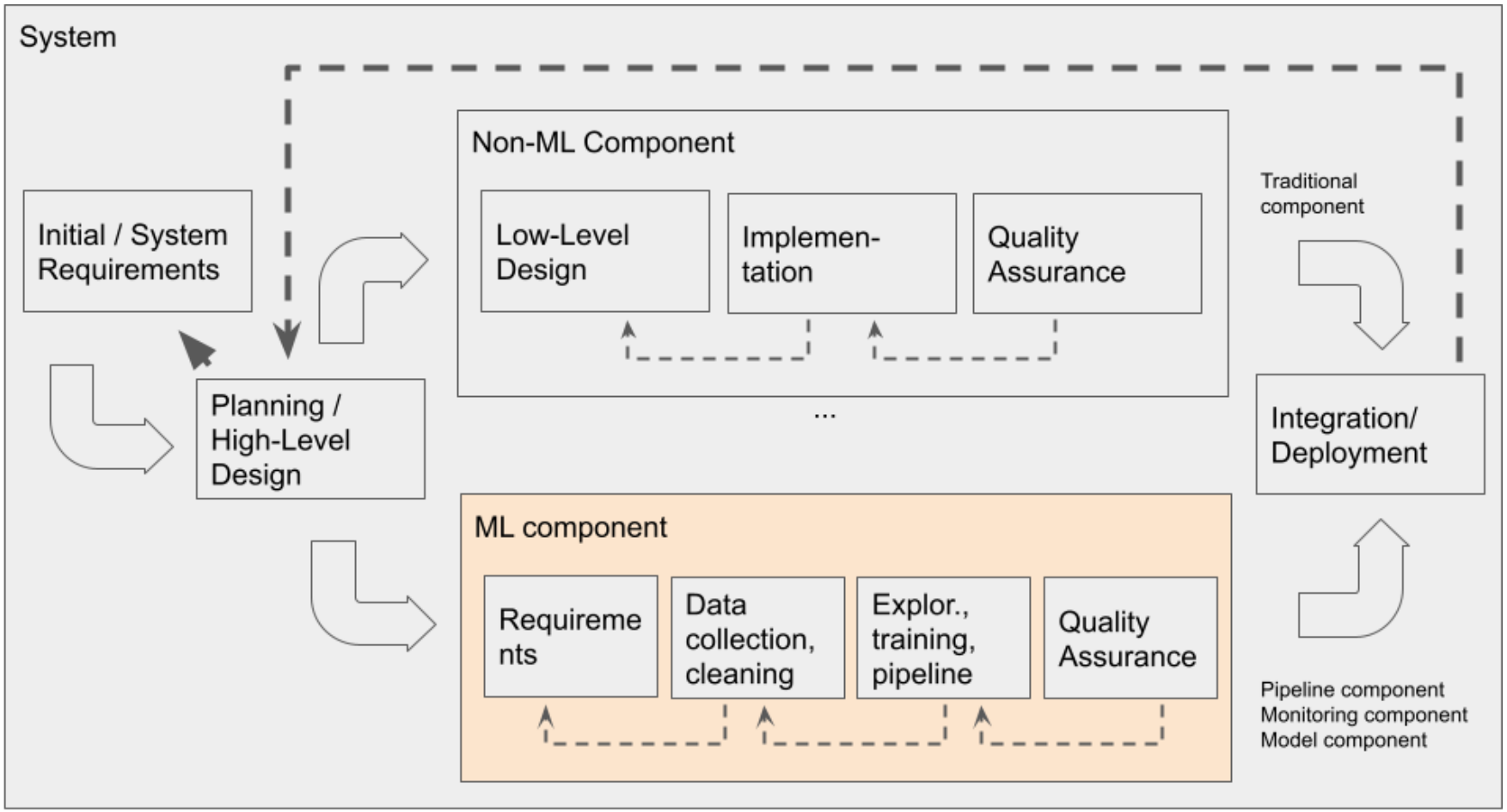
(CRISP-DM)

PROCESS MODELS

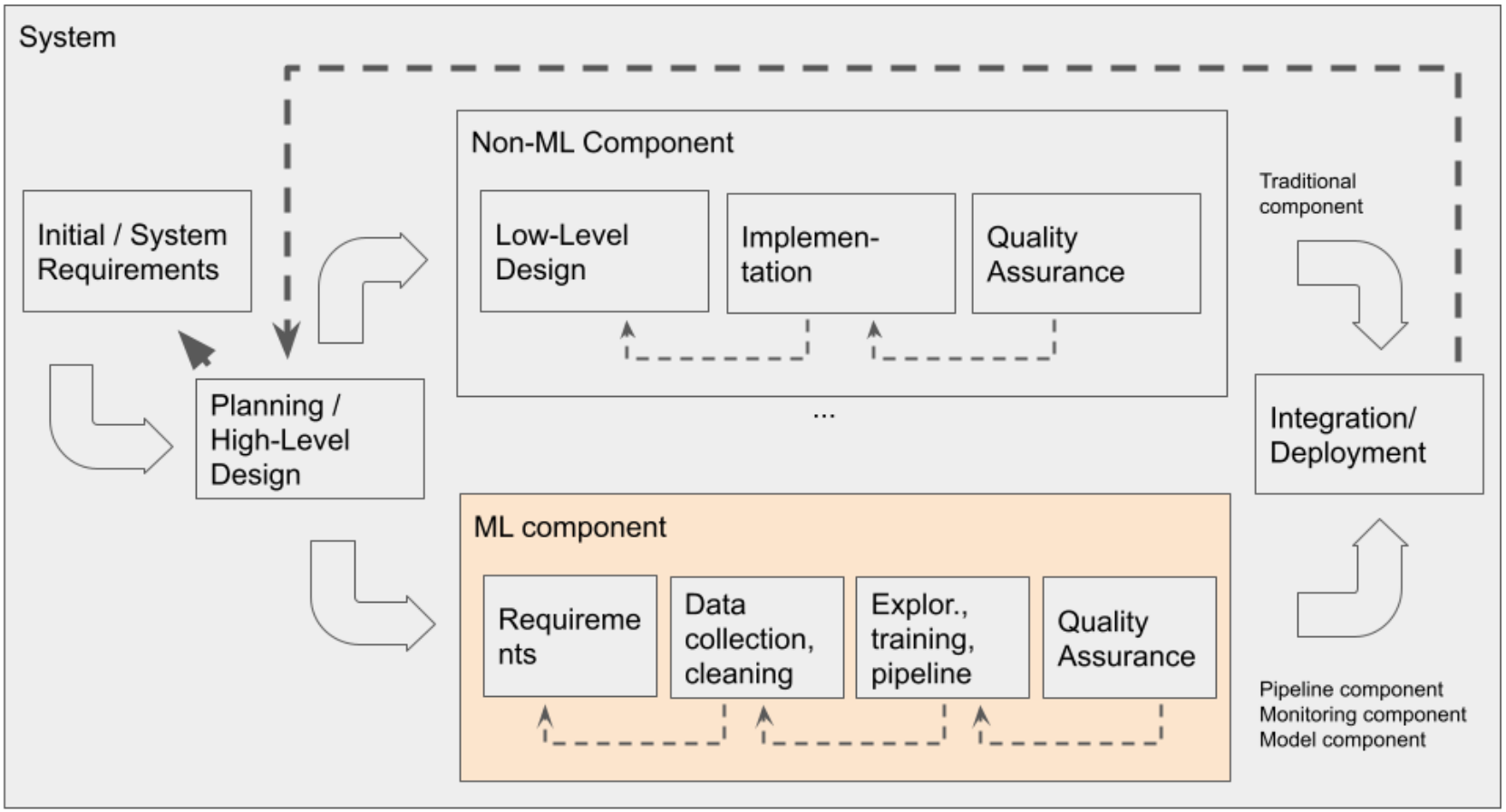




INTEGRATED ML/NON-ML PROCESS



DEVELOPMENT TRAJECTORIES



Upfront design? ML first? Non-ML system first? Incremental refinement?

SUMMARY

- Adopt a whole-system perspective on ML in production
- Design for quality, don't patch it in later
- Understand qualities and tradeoffs, including performance
- Consider implications of qualities on non-ML components
- Build interdisciplinary teams, integrative process

Interested in talking about your experiences in interdisciplinary ML teams? Contact us.

TRADEOFF: MODEL IMPROVEMENT VS SYSTEM SAFEGUARDS