Operationalizing Machine Learning in the Enterprise

TDWI / BARC
Munich
June 20, 2022

Mark Madsen

Introduction

There is a lot of advice in the industry – most of it wrong – or simply lacking the context that helps you apply it.

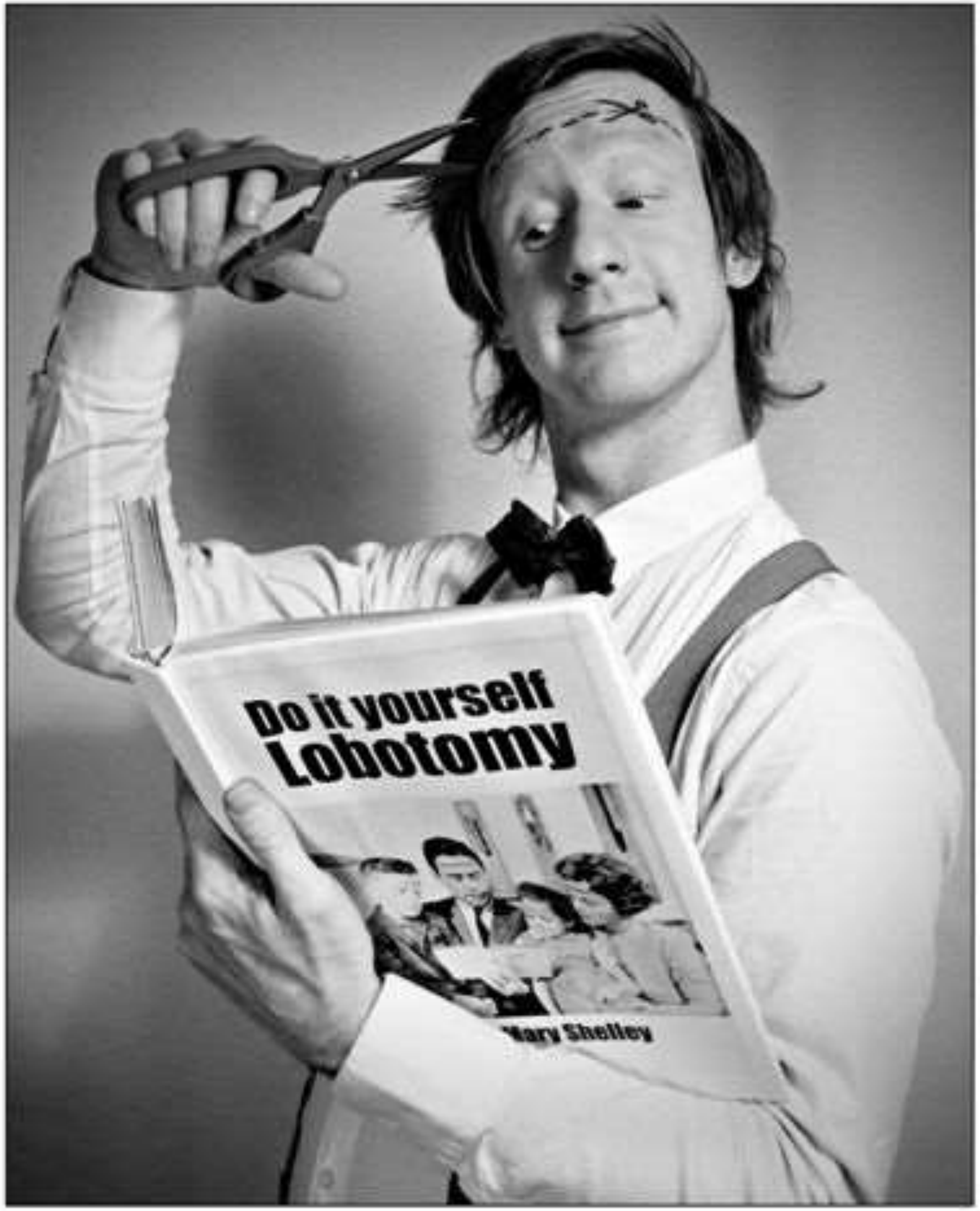
What are the assumptions made about data science usage, goals, requirements, practices?

This presentation will not

- discuss much about application development or software – this is about operations, not technology

However, it will

- cover foundational concepts, observations, and practices in the industry



What do we mean in this session by operationalizing ML?

No:

- Simply moving a project into production.
- A project office with many projects doing their own work in isolation.

Yes:

- Building a data science program that provides a repeatable data science capability to the organization.
- Managing the full lifecycle of ML uses and systems

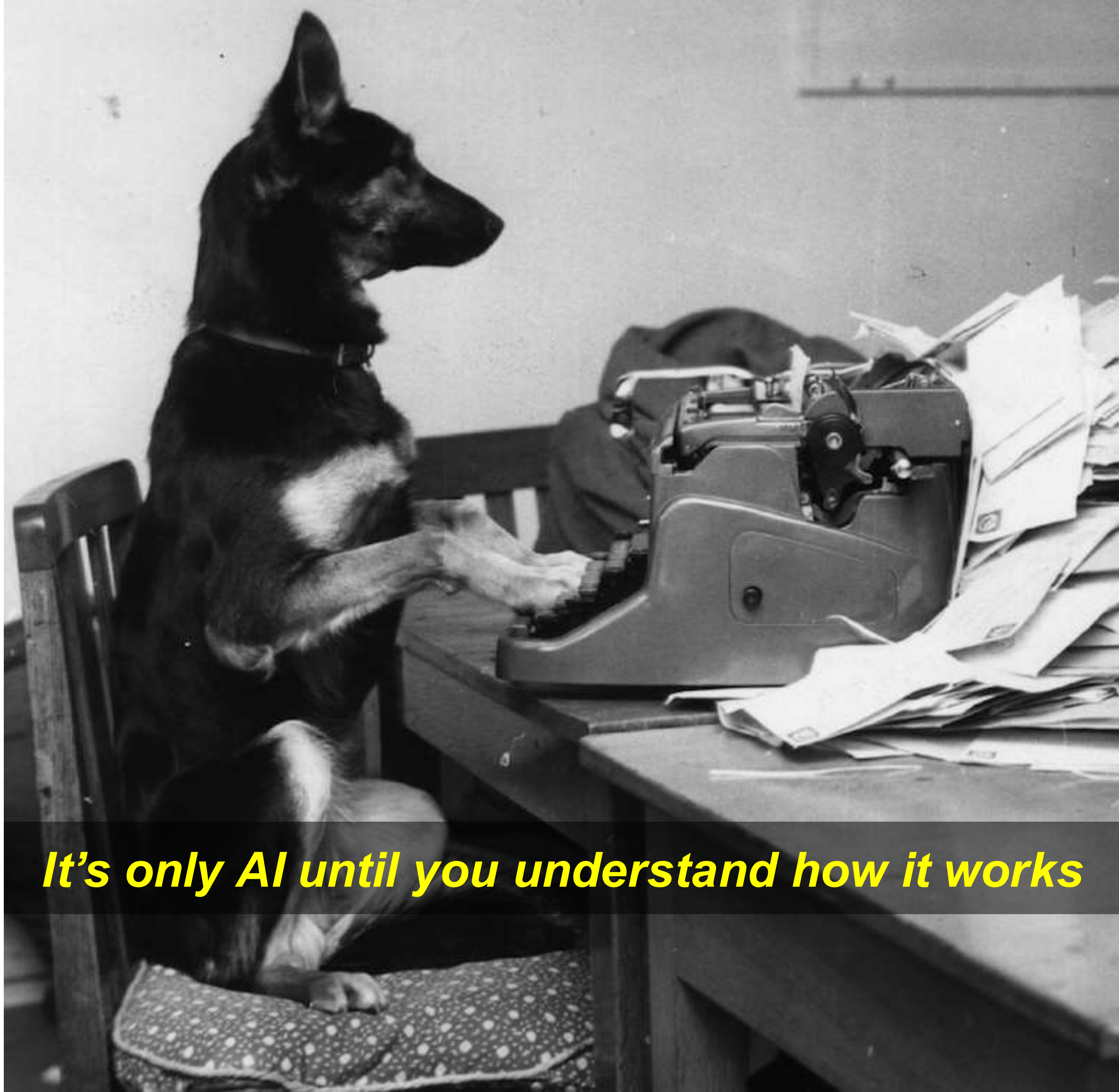
But first we talk concepts, then about process and org

More context for us

Data science is a set of practices and techniques used to draw insights from data, and aimed at a specific a goal.

Machine learning is a set of techniques that fit a model to data, i.e. “learn” a model. The input data becomes code.

AI uses ML and other techniques as part of a larger system that takes actions.



Why this topic? Some numbers

Reality behind the Hype

65%

of predictive models are never implemented in production

“Build an ecosystem that includes not only tools but also data, people and processes”

5 months

Average to develop, test, validate, deploy and scale one new analytical model

“Acting like a Fintech is a lot easier said than done”

25%

of data scientists time is lost in interactions with development teams

“The manual back and forth during development results in costly delays”

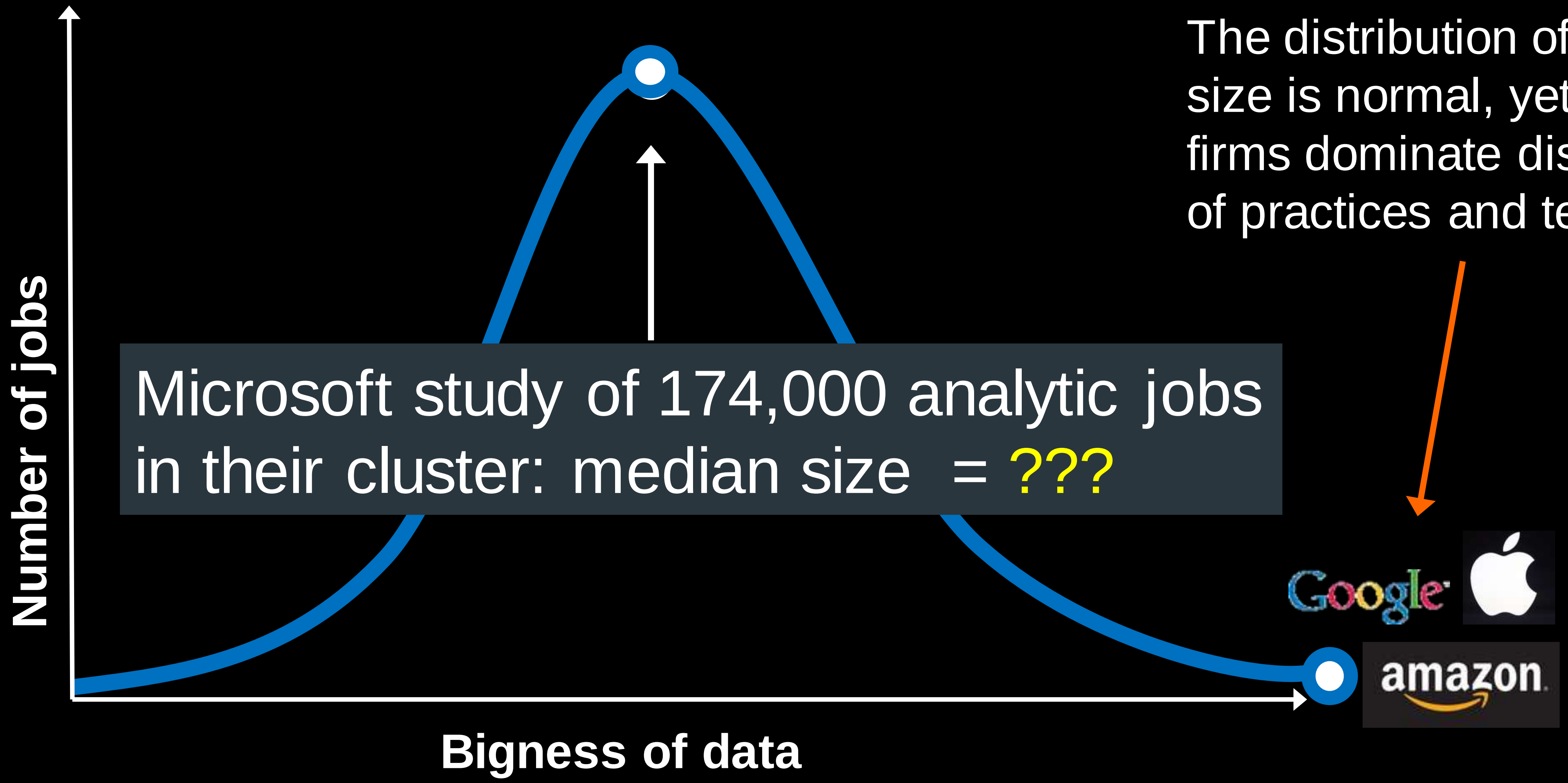
Operations lies at center of any organizations’ enterprise ML or AI strategy
If you can’t sustain it, you won’t benefit

Sources: Gartner, PwC, Accenture, McKinsey, Teradata

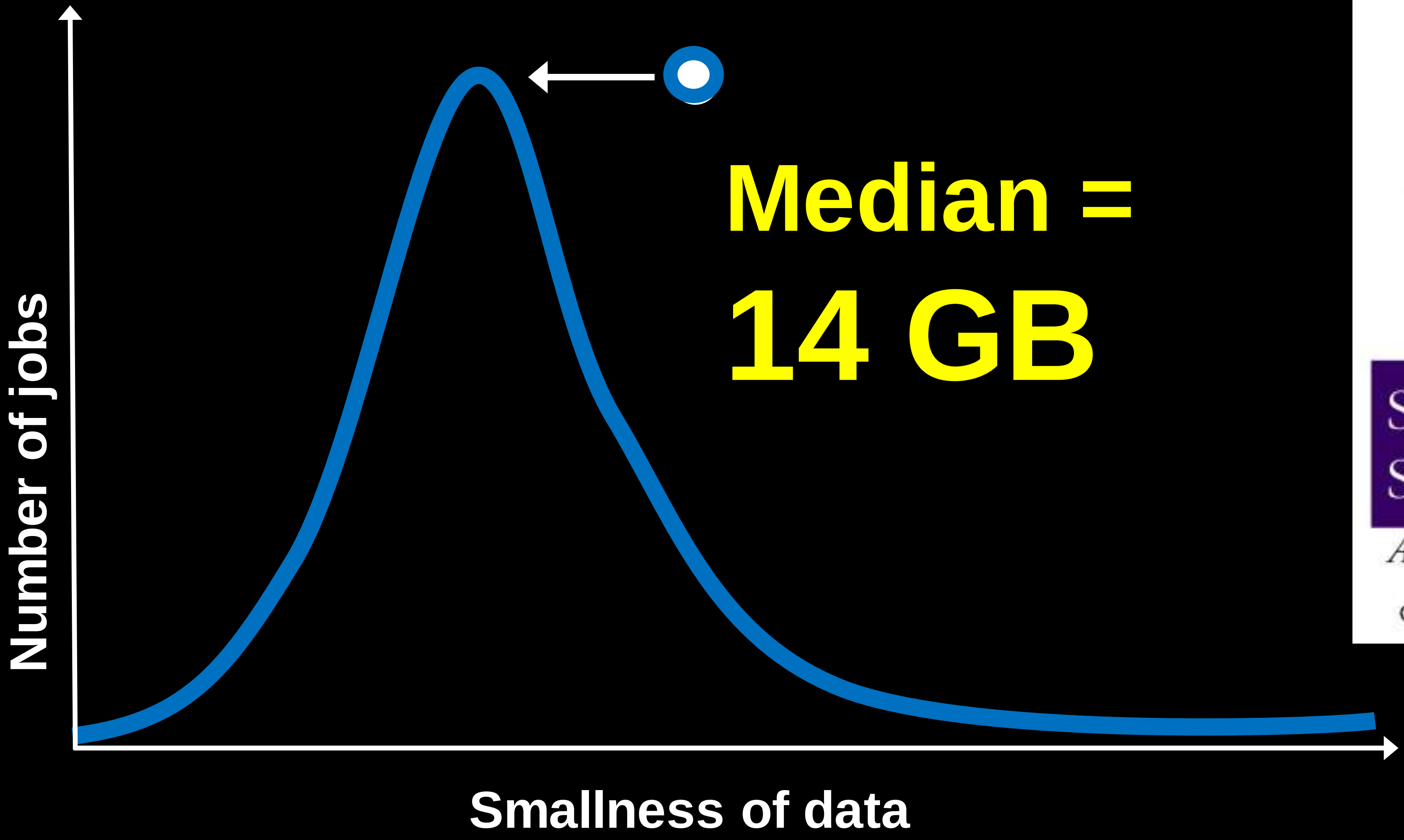
Why do we see these failures?
Must be the technology, so
buy newer technology!

Do you need new technology because of scale?

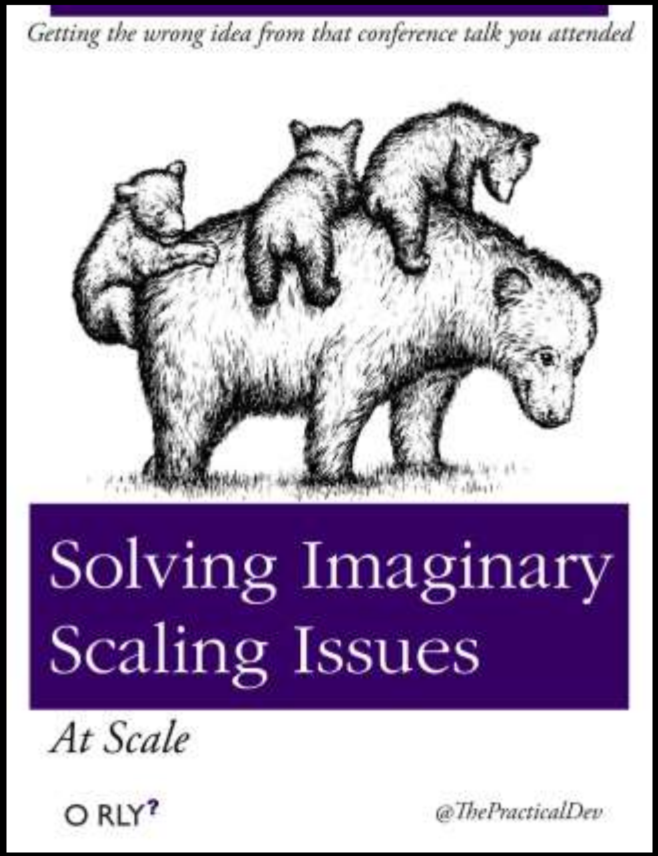
The distribution of data size is normal, yet these firms dominate discussion of practices and technology



Analytic datasets are usually not big



Median = 14 GB



"It's a poor carpenter who blames his tools"

The technology is not to blame. The people who chose to use it are to blame.

Tools are made to fit a job, not the reverse.

Therefore, the technology is less interesting than how and why it was chosen so…

<u>What are you building with it?</u>

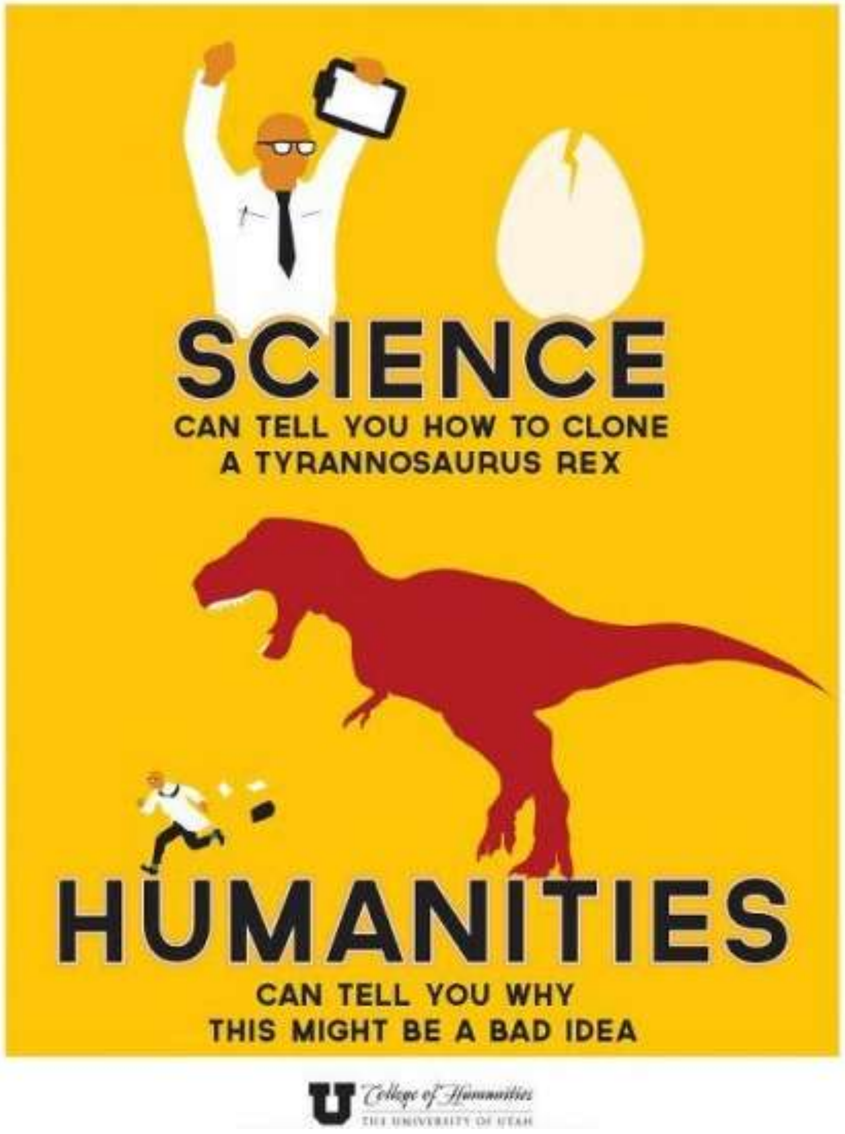
What is the nature of the operations problem?

“Begin with the end in mind”

The starting point *can’t be* with technology. That’s like buying bricks when you want to design a house. You may get lucky but…

The goals and specific uses are the place to start your work

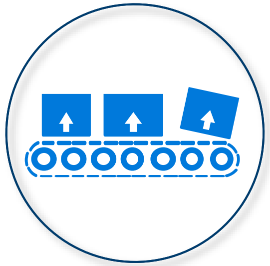
- Use dictates the needs
- Needs dictate the required capabilities
- Capabilities are delivered by technology, process, and organization



This is how you avoid spending €20M on a Hadoop and spark cluster in order to serve data to analysts whose primary requirements are met with laptops and a query.

First question, what is your *organizational* goal?

To complete an ML **project**?

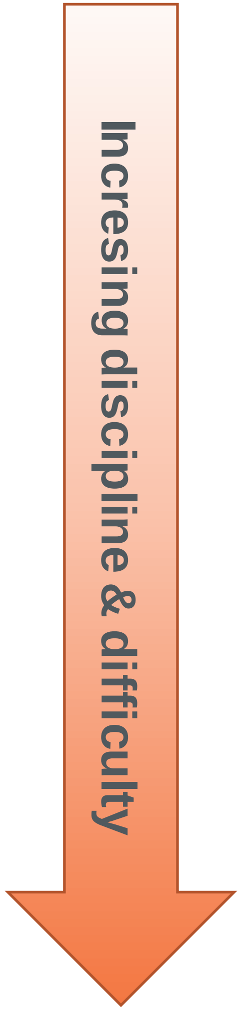


To create a data science **program**?

To build and maintain an analytic **product**?

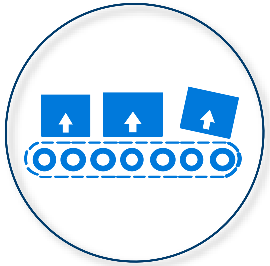
What you need to complete a project is not the same as what you need to sustain a program is not the same what you need to maintain a product..

Each of these has different assumptions, conditions, and practices



To complete a project?

Clear goal and requirements, independent, time bounded, single budget, blended team of dedicated and part-time staff, no or limited infrastructure, varied practice.



To create a program?

Multiple overlapping projects, open-ended, interdependencies, focus on repeatability of activities and organizational effectiveness, multiple budgets, shared infrastructure (constraints), requires shared practices

To build and maintain a product?

One organization with dedicated staff and roles, lifecycle, tighter dependencies, operational budget, focus on product-market effectiveness, dedicated infrastructure, requires cohesive process

Programs and products drive the need for infrastructure

Note: Don’t let tech vendors confuse you

Is your business model and IT infrastructure really like these companies? Is your project?

Should you take advice from someone who hasn’t experienced the problems that you face?

The market and ML products

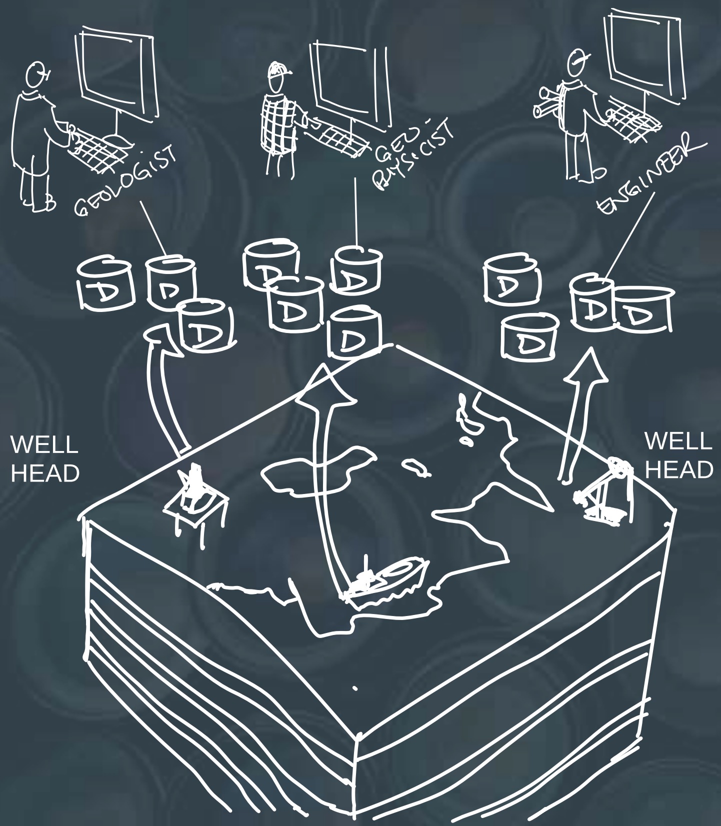
ML / analytic products are primarily found in digital companies

Digital companies are not like the typical enterprise

- Business and infrastructure assumptions are inverted
- Less varied and less complex data
- Uniform technology environments and practices
- More critical requirements (% of revenue threats)
- Dedicated staff and roles
- Focused product goals and orgs

Study the problems they are solving, and the context in which those problems are experienced

A common approach to data in data science organizations



Example use cases

One-Pipeline-Per-Process

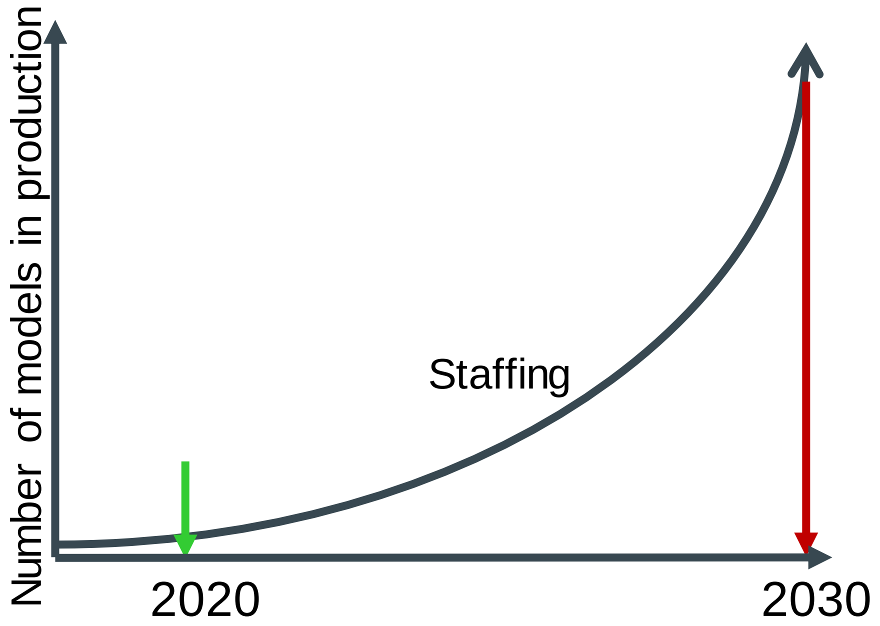
Redundant Effort / Cost / Complexity / etc.

Project approach to models in production at org scale?

Most organizations have a project-based approach. This makes it easy to deliver new projects. Independence and flexibility.

With the silo/pipeline approach:

- If each model takes X% of effort to maintain, how many models can you build before you use up 100% of your time?
- Automation helps, a little.
- Efficiency helps more.



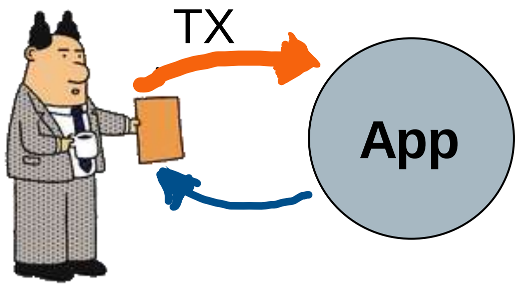
The projects-as-silos and pipeline approach will not work when running models in production at the massive scale required for total automation

What is the nature of the system(s) you will build?

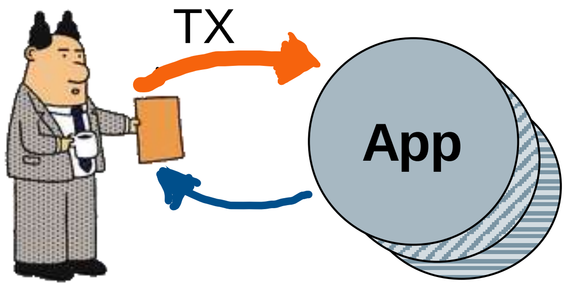
Second question, what type of ML system are you building?

Which means we need to talk about categories

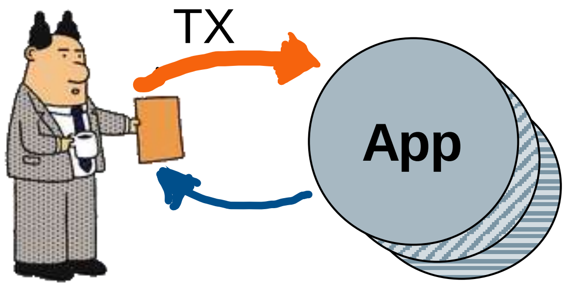
How an organization works



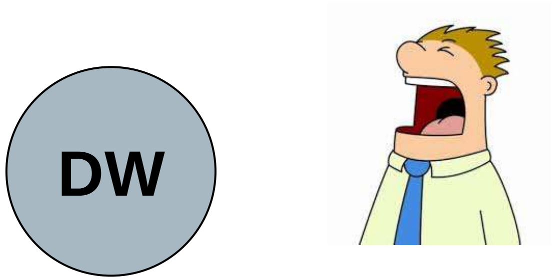
Many applications, many activities, many users/customers



How do you see the full picture across all of them?

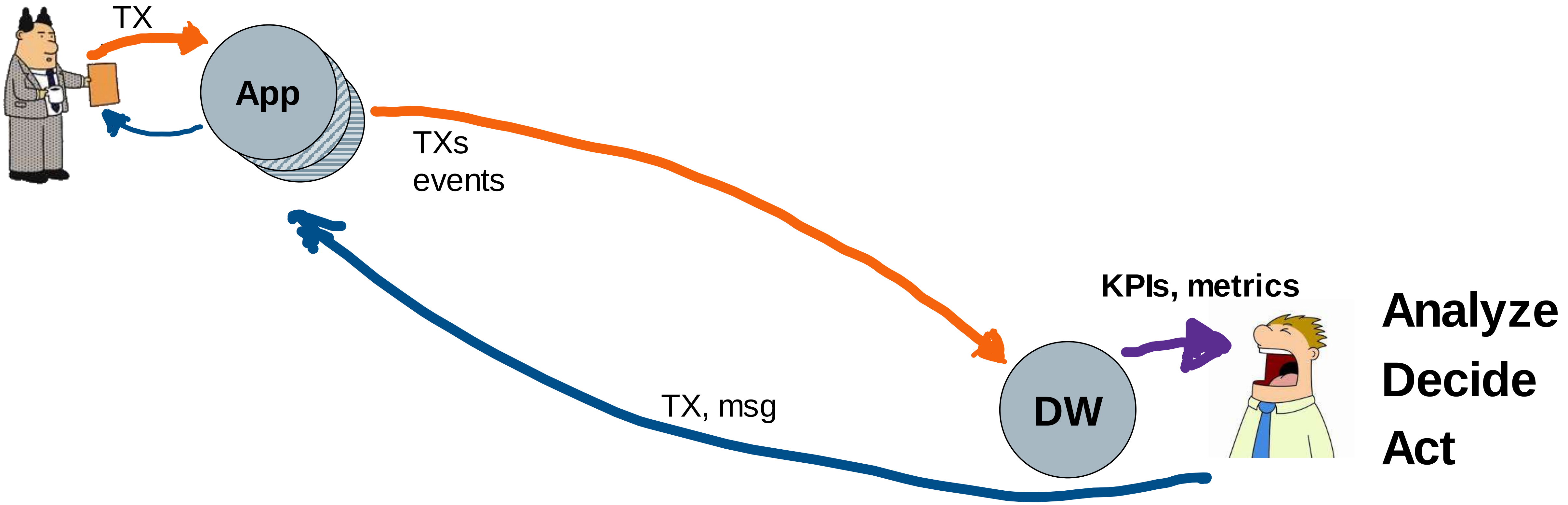


What does a manager do when there's a problem?
One report from each application isn't a sustainable answer



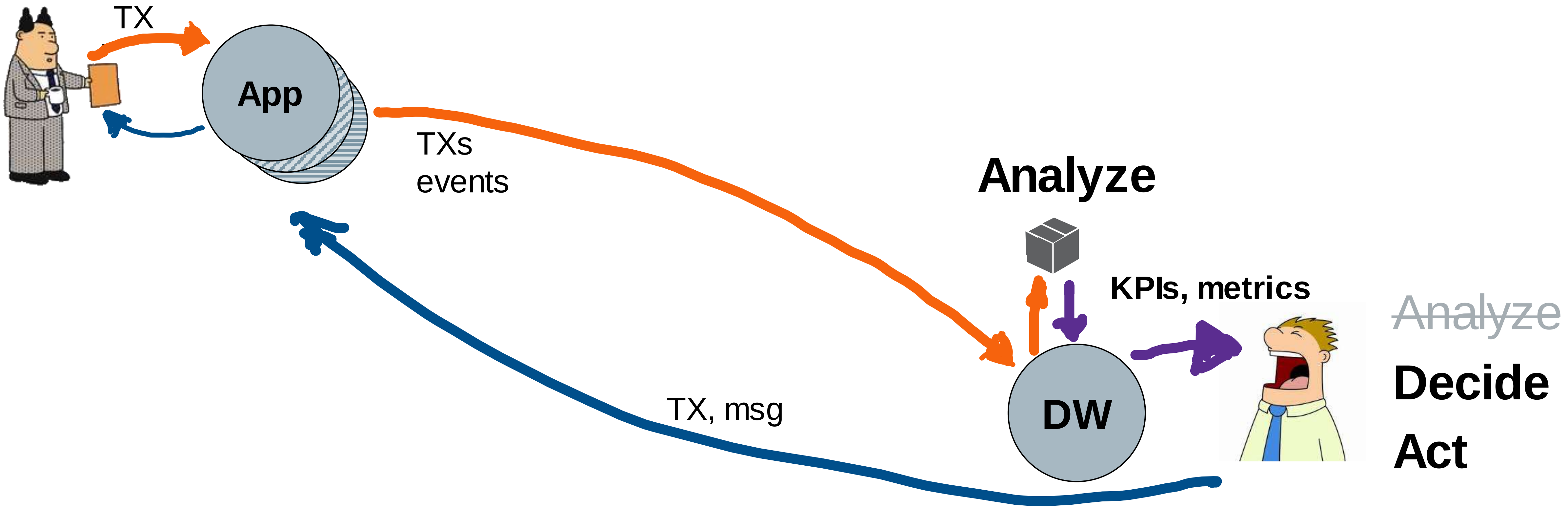
This is the “decision support” type of system

The purpose of this type of system is to provide information to someone who will use it to take action. The goal is to make decisions, not get reports



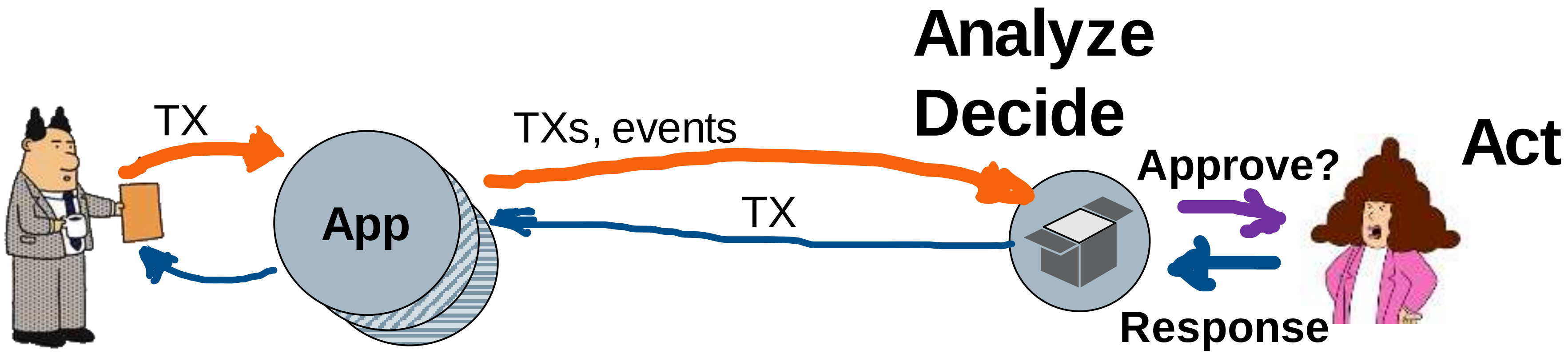
Both batch and streaming analytic models have been running within this type of system for decades

e.g. customer segmentation, churn, fraud, response modeling
Usually batch, ran from data extracted for metrics, often pulled data from a mart/DW and fed data back into the system for use.
A manager or user still makes decisions based on the information



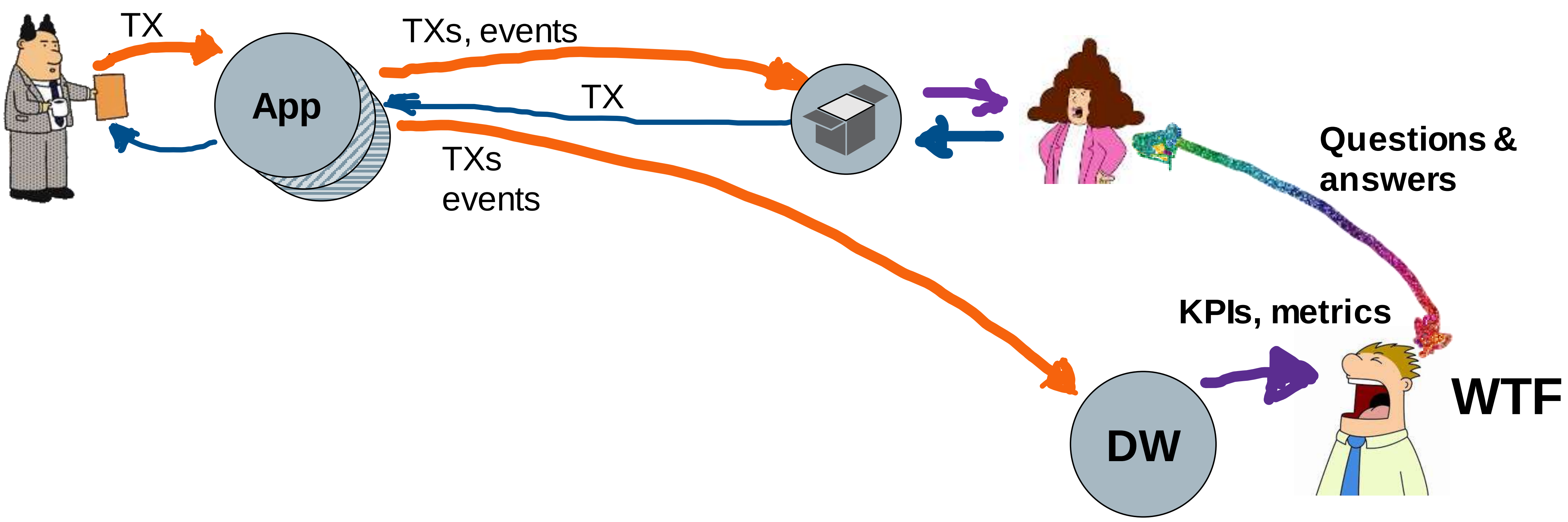
Human-in-the-loop systems are different than decision support

Applying analytic output within a process context
Machines gain agency, humans lose it; ability to act is curtailed
Examples: purchasing changes, call center upsell/cross-sell recommendations



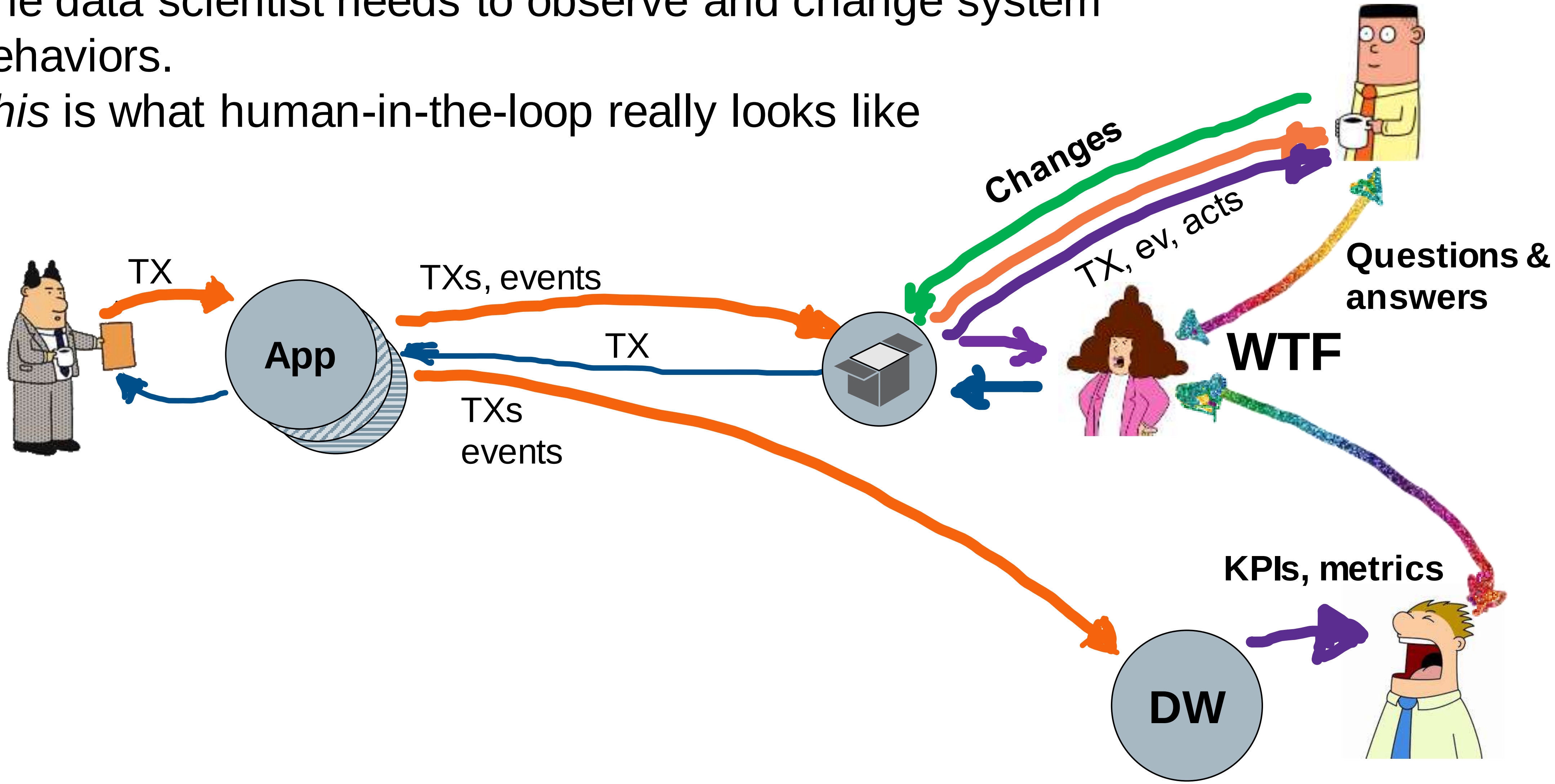
When there's a problem, the resolution is a conversation

The results of the model's use should be visible via the KPIs and metrics.
People call people to see what's going wrong.
This is one reason to have a mature BI practice before doing machine learning

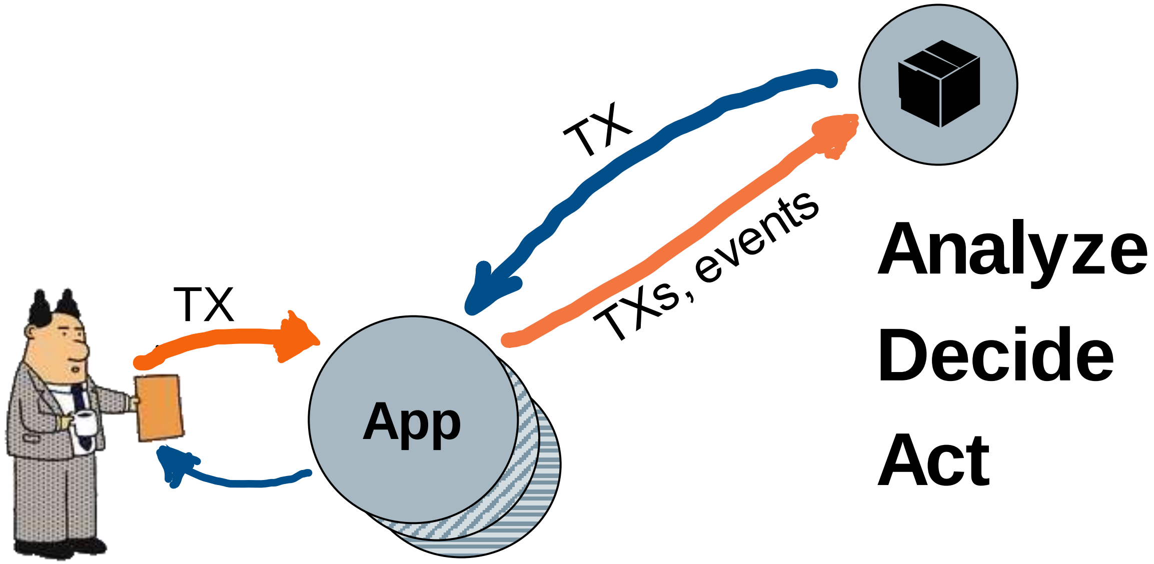


Somebody built the models – the data scientist

More communication is sometimes needed for a resolution.
The data scientist needs to observe and change system behaviors.
This is what human-in-the-loop really looks like



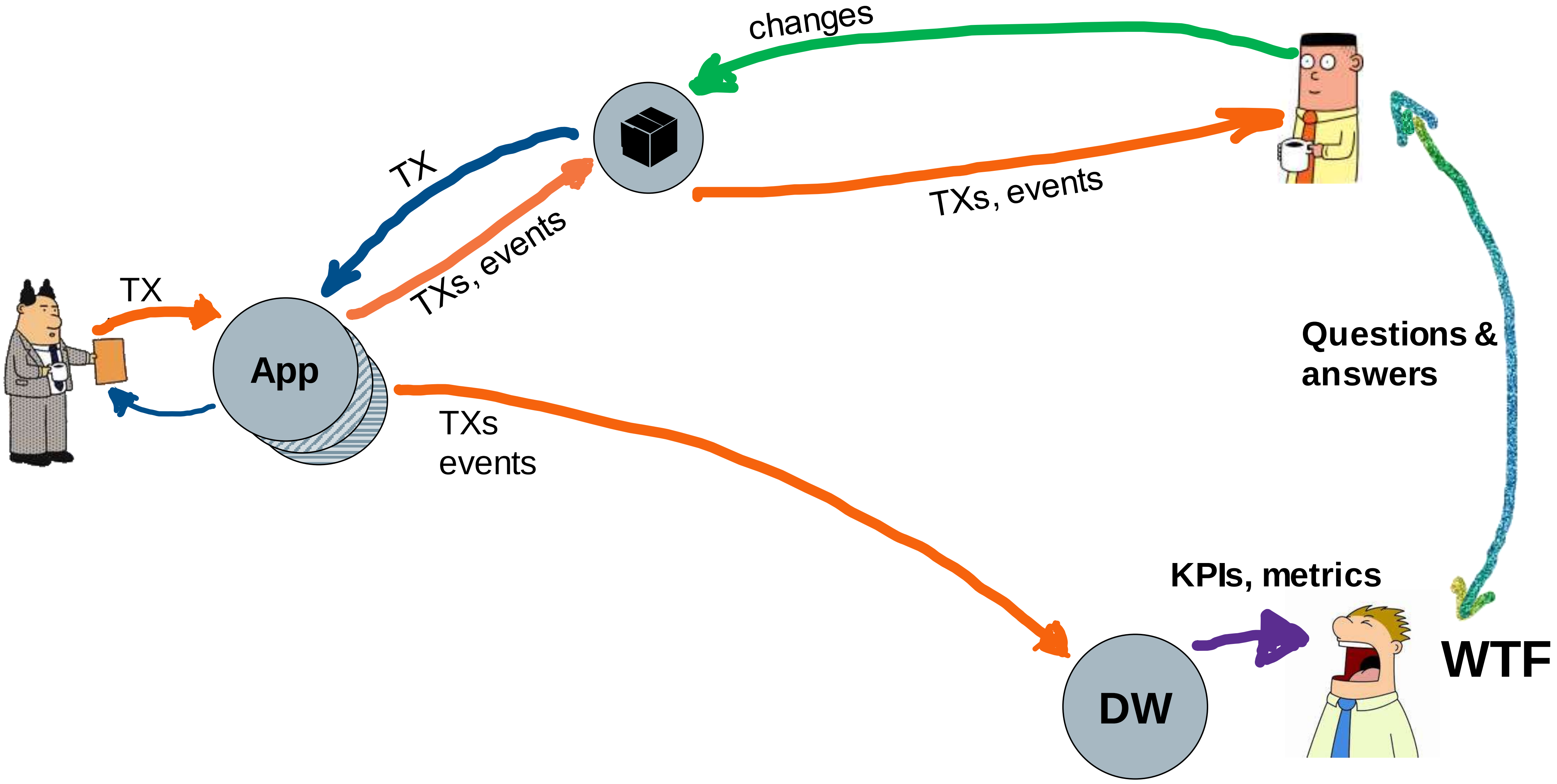
Enter black boxes – the “embedded ML” system types



They run independently, from batch latency to faster than a human can perceive.
Human agency is removed from the system as it executes.

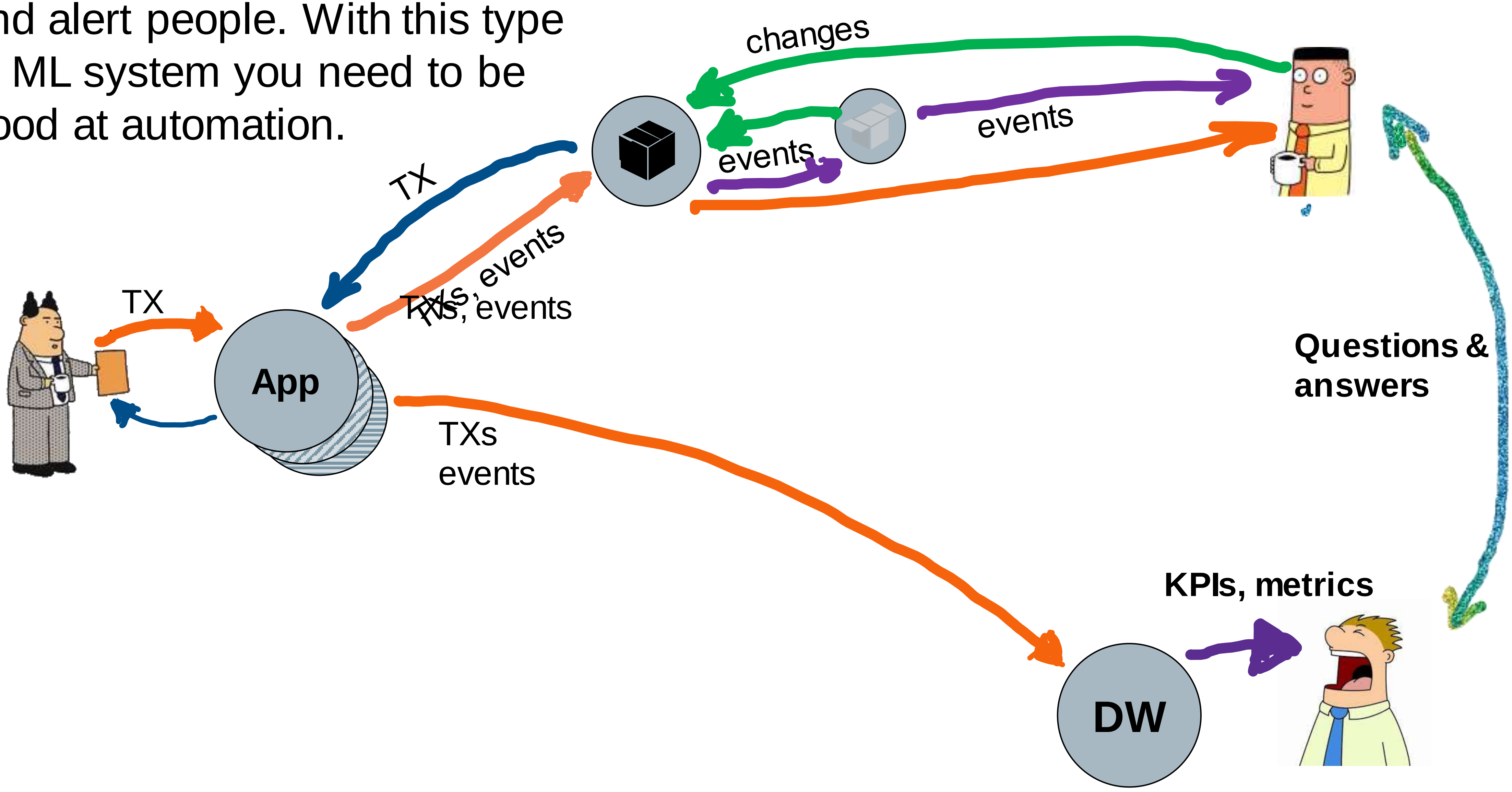
Dirty secret: there is always a hidden human in a loop

Black boxes still need oversight because things go wrong



Black boxes beget gray boxes because speed is needed

Gray boxes control black boxes, and alert people. With this type of ML system you need to be good at automation.



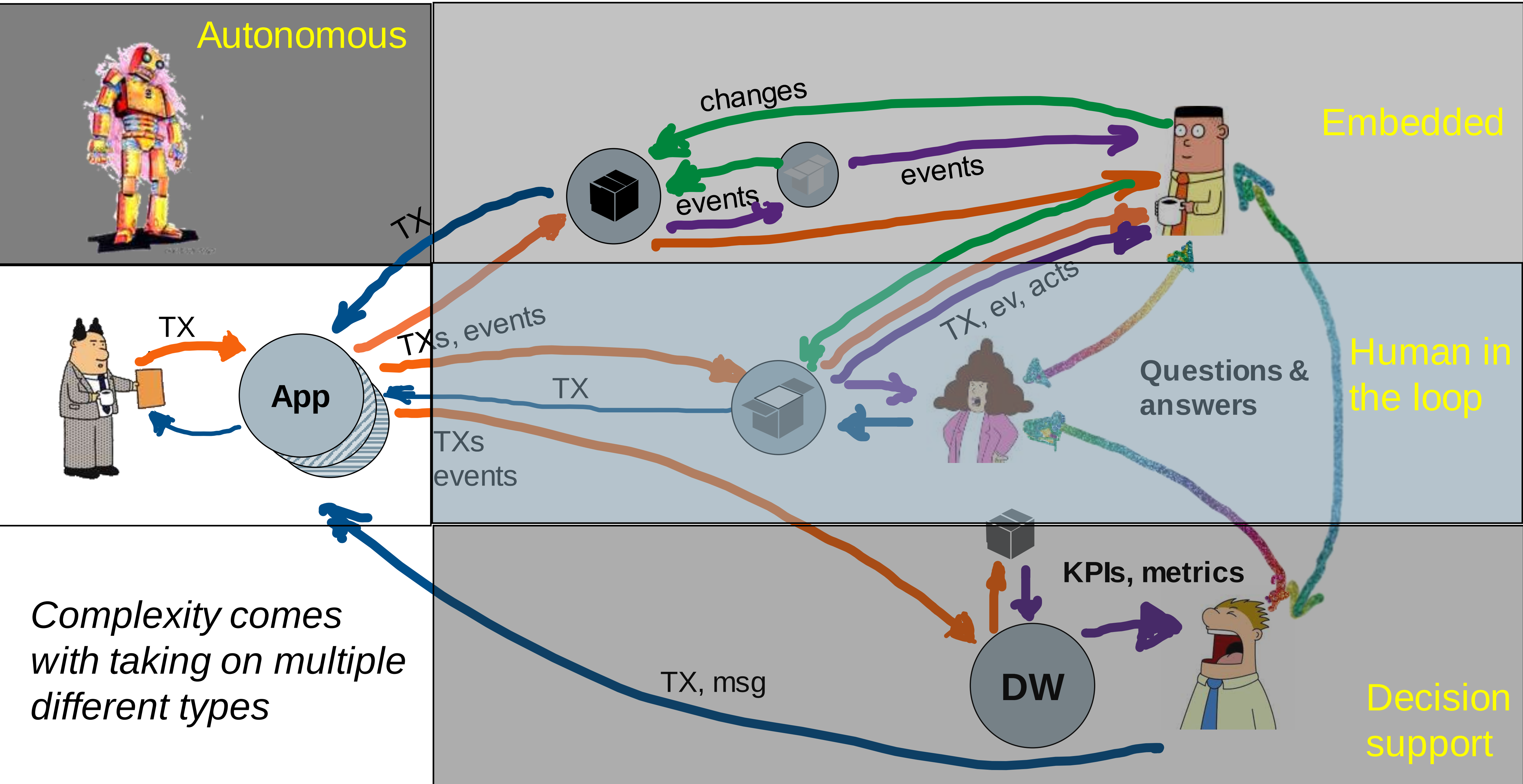
Autonomous systems are a fourth type of system

Most ML is embedded in a single, central system. Autonomous systems operate with no direct central control.

Governing a model and it’s use is *far* more complex when the system is not a centrally controlled service.

There are more complex internal controls, even more telemetry and introspection, and a great many more failure modes that can increase risk.

As a program matures, you could have all of these at once



Complexity comes with taking on multiple different types

How will you use the system you build?

3rd question, how is a system being used?

How you use a system defines how you approach design of the system, and the analysis of needs

Augment: work to support a human role

- Emphasis is on human factors

Automate: replace human involvement

- Emphasis is on integration

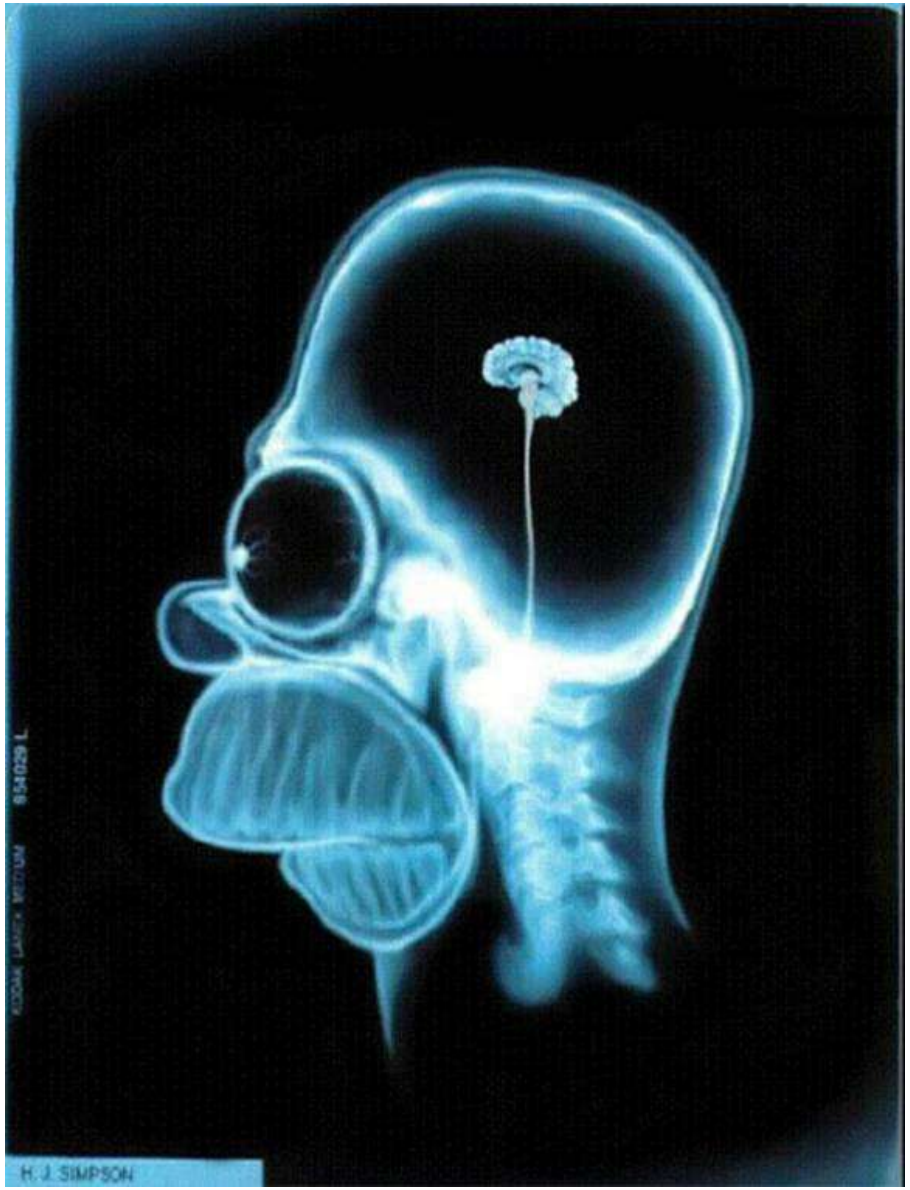
Invent: do something new that has not been done before (in your org)

- Emphasis is on experimentation and learning

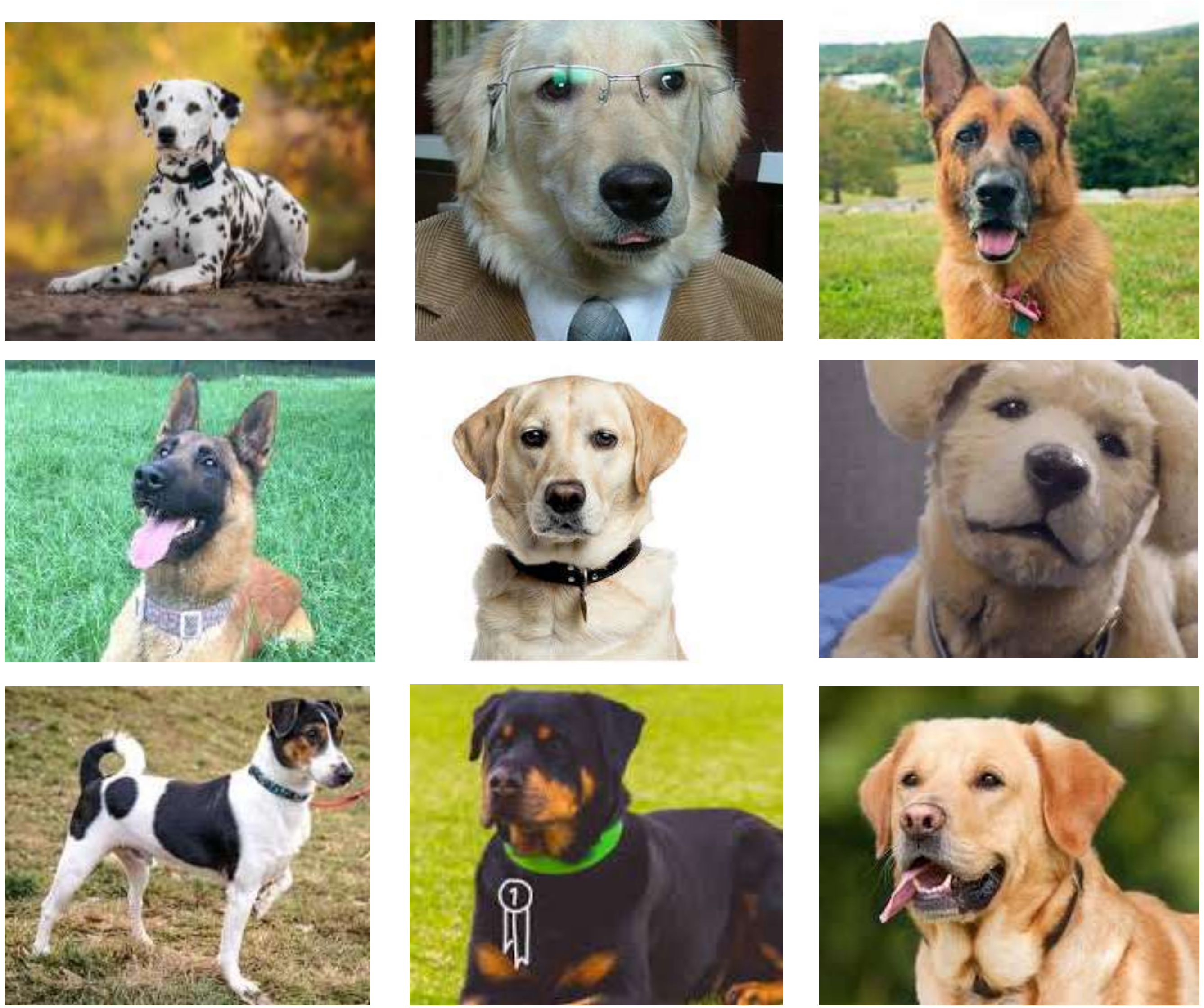
Each one takes a different approach, and a different team composition.

Replacing people with AI for a task is less likely to succeed than either helping people in their tasks via ML augmentation, or doing new things that people never did.

For example, using binary classification and computer vision

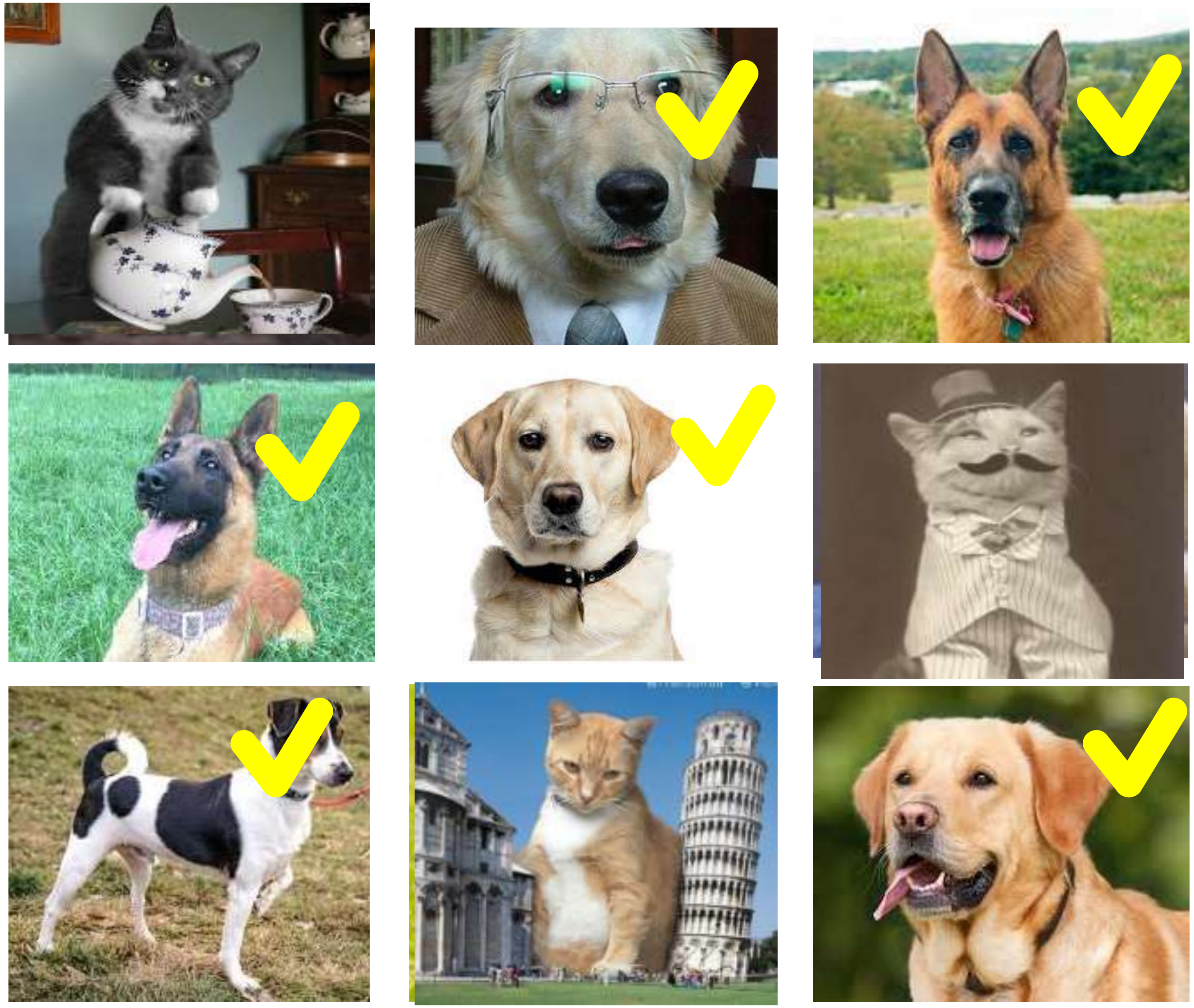


AI image processing: detect dogs. Here’s the training data



Dog or Not Dog?

Dog or Not Dog: 100% accurate! But why?



“The model learned via the shortest path through an N-dimensional space to find the proper classification”

Diagnose X-rays using computer vision: 100% detection rate

The <u>AI shortcut problem</u> matters:

How medical testing really works:

1. You might have cancer
2. Get a retest at a second lab
3. Only likely candidates go to second lab
4. All positive x-rays are used for training
5. If the x-ray came from lab 2, you have cancer, 100% of the time (in the data)

Too good = wrong

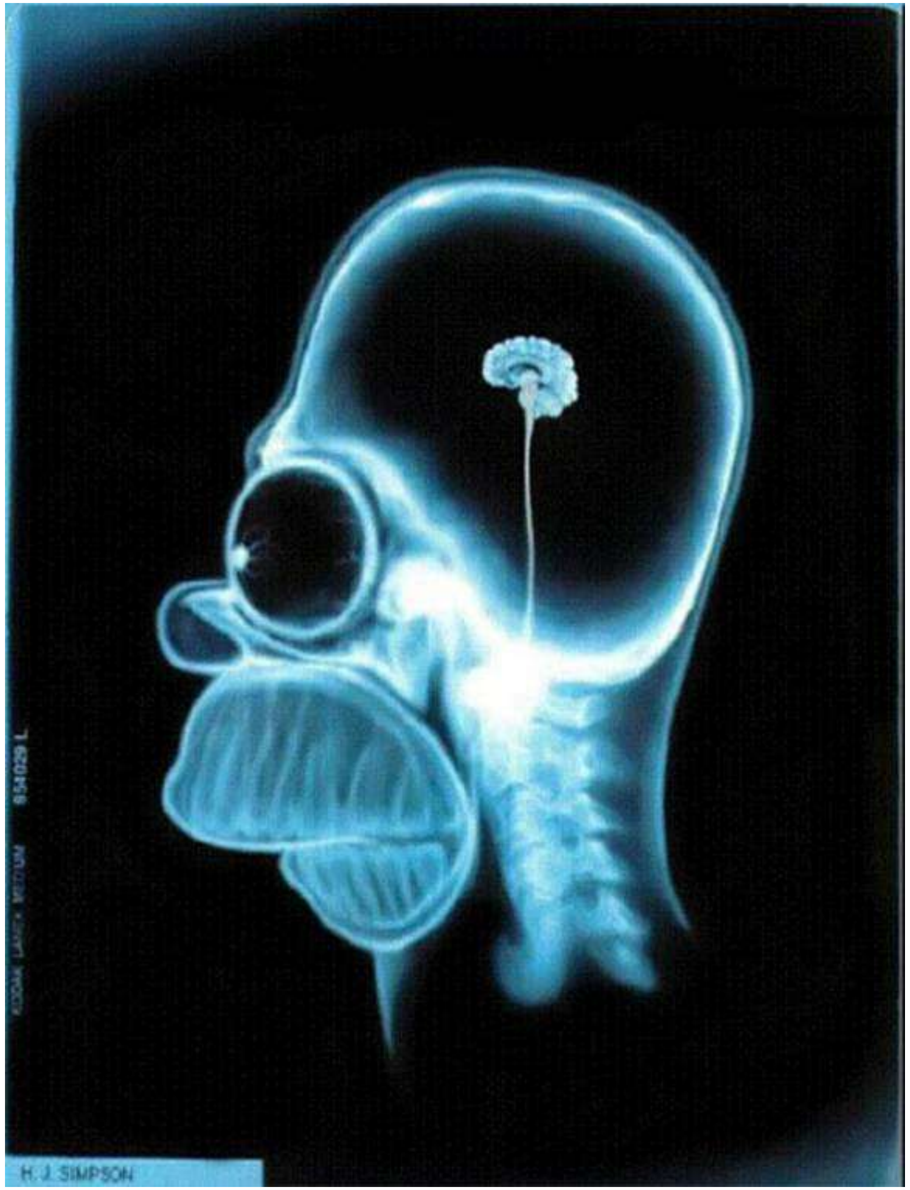
AI really learns <u>datasets</u>, not the problem domain

Beyond toy examples, this problem matters

AI learning and reasoning challenges will limit AI applicability until they are solved.

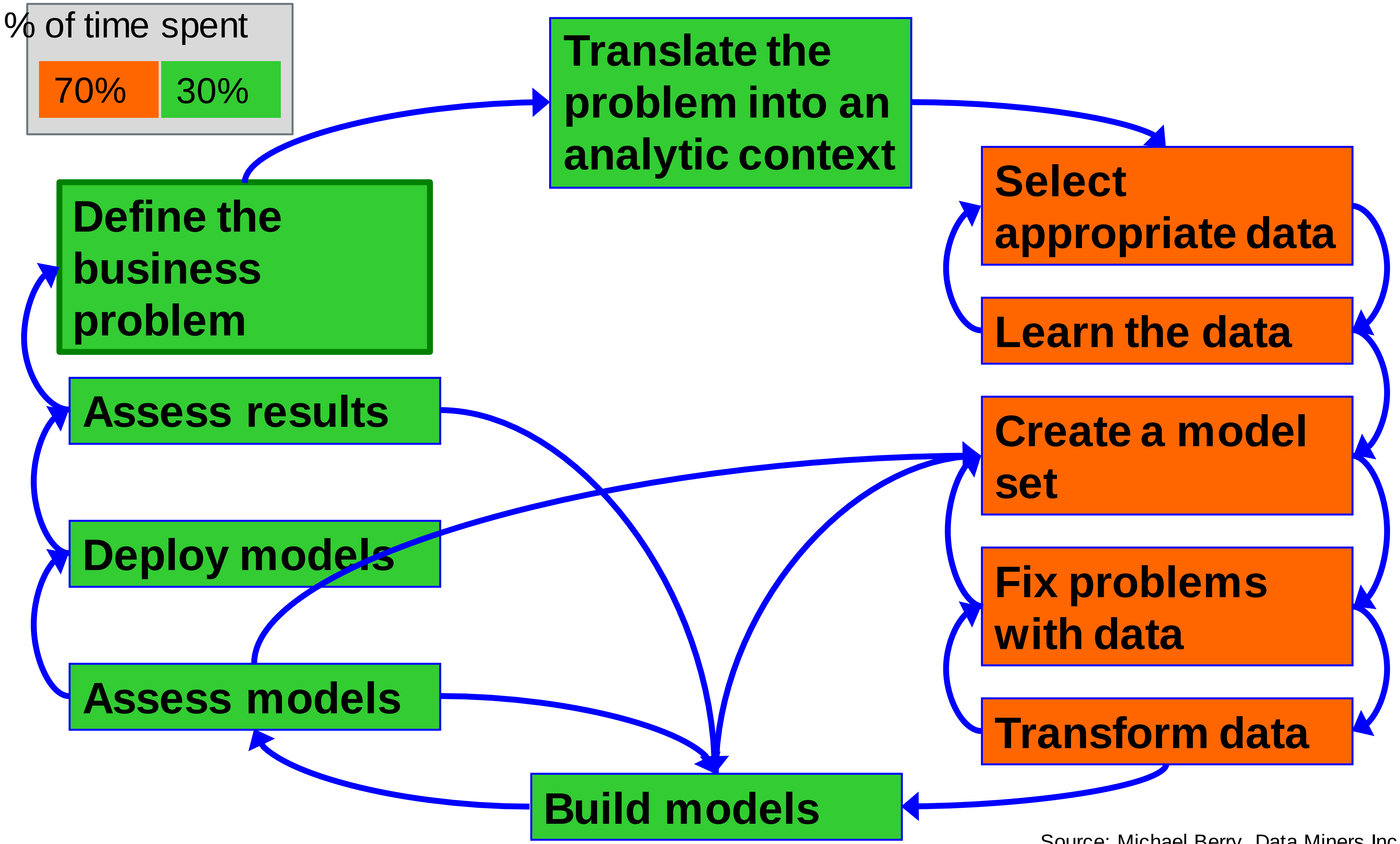
The uncertainty and risk means your approach should be to augment people, not replace them with AI

Augmentation and net-new uses with AI are two areas where open world problems can be contained

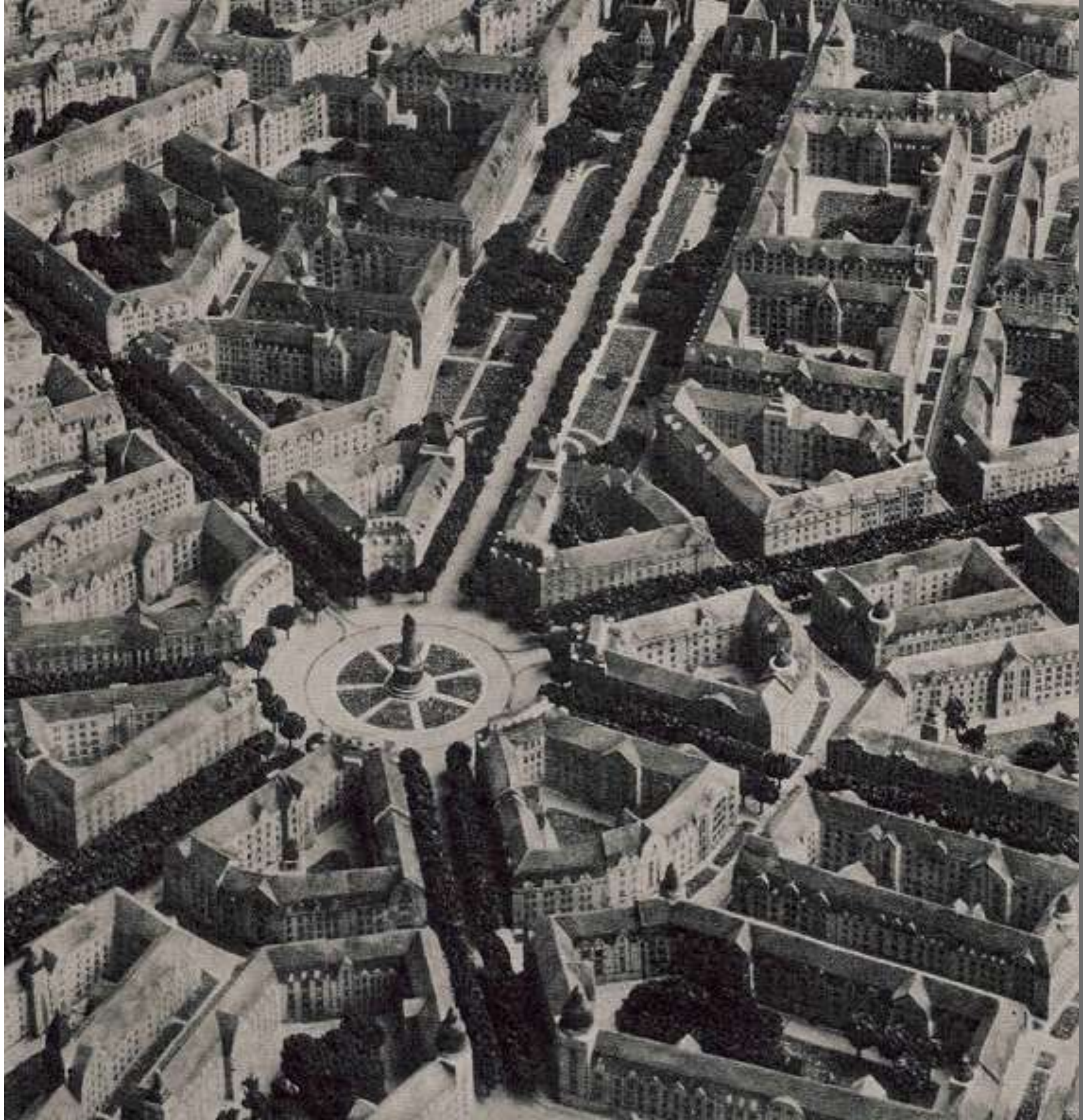


How do those systems fit into operational workflows?

Starting with the process everyone does: building stuff



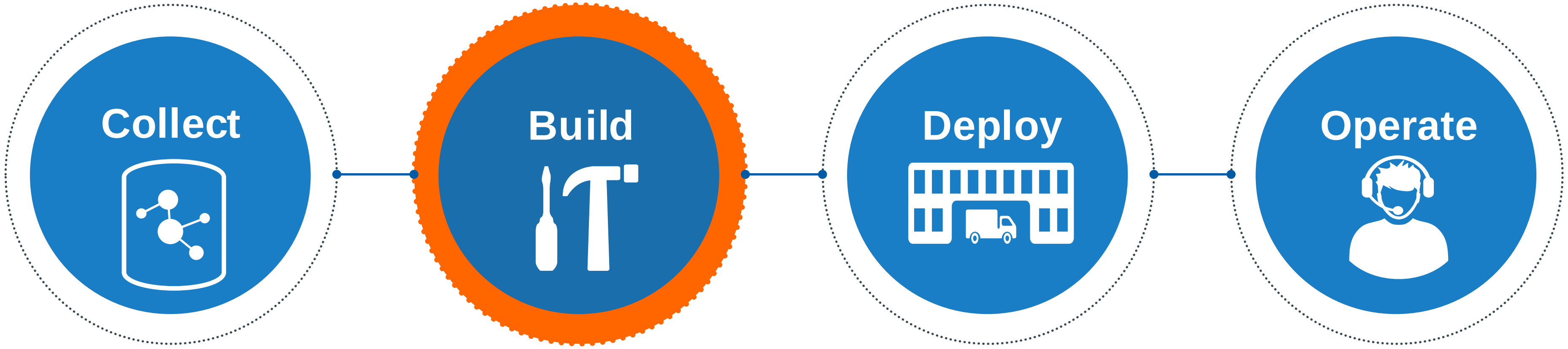
Source: Michael Berry, Data Miners Inc.



"Always design a thing by considering it in its next larger context - a chair in a room, a room in a house, a house in an environment, an environment in a city plan."

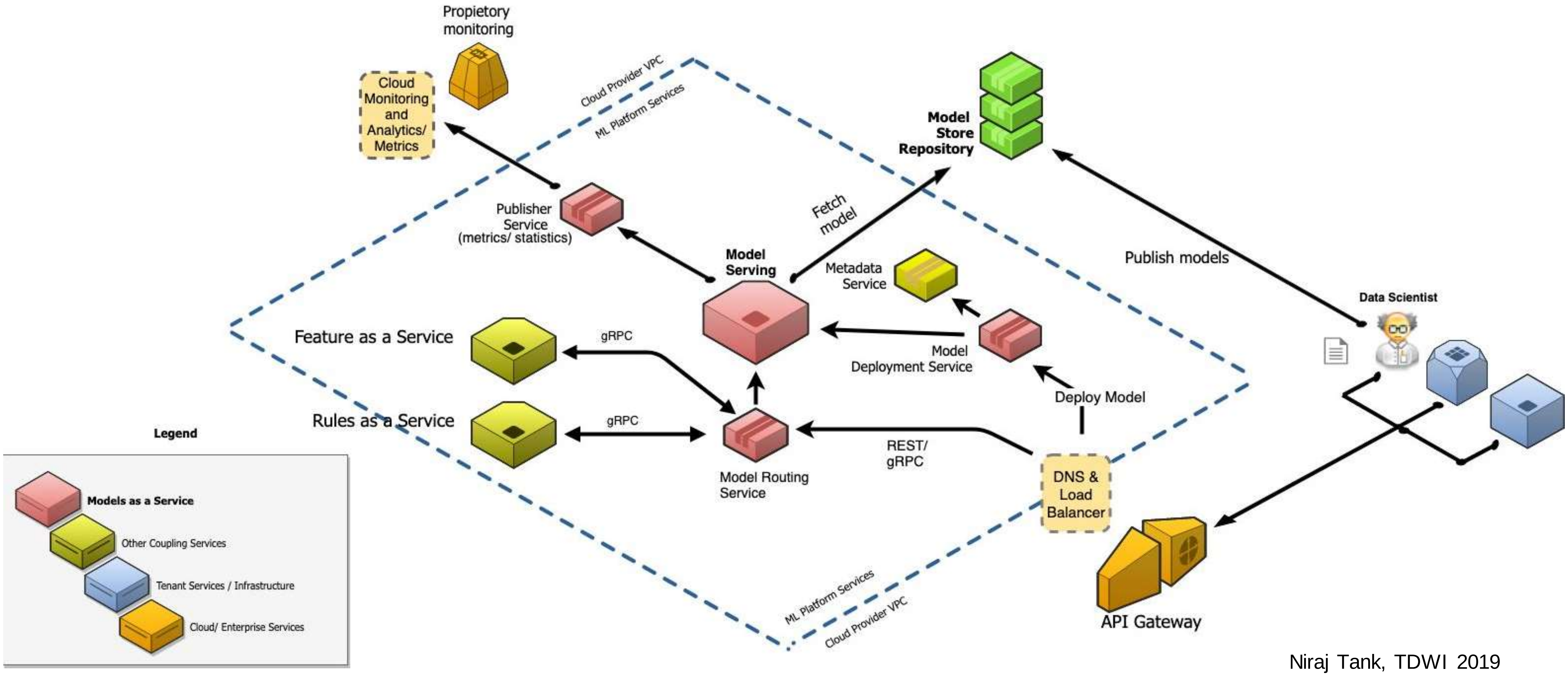
– Eliel Saarinen

Expanding the perspective beyond the initial bit



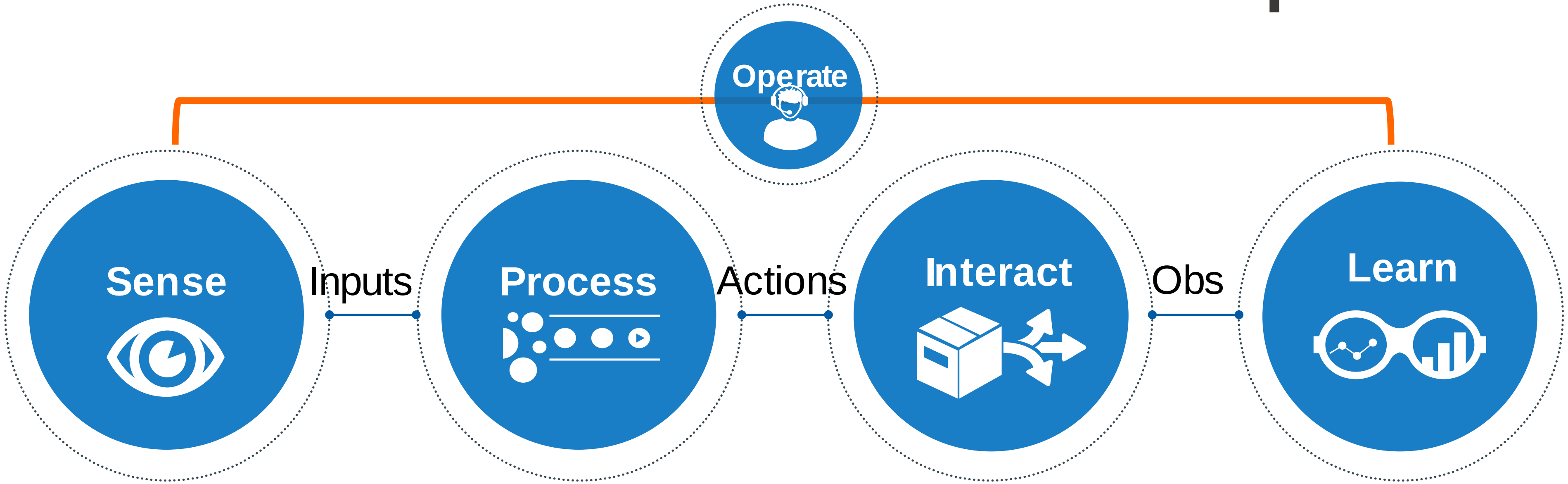
- There are upstream parts to the development process: collecting and managing data, both for dev and in prod.
- There are downstream parts, in deployment and then in production operation.
- Data and artifacts are exchanged as part of the workflows

One implementation of an ML publish-deploy-track-invoke-monitor process, from the point a model is approved



Niraj Tank, TDWI 2019

The execution workflow itself is complicated



Collect inputs
AKA clean data

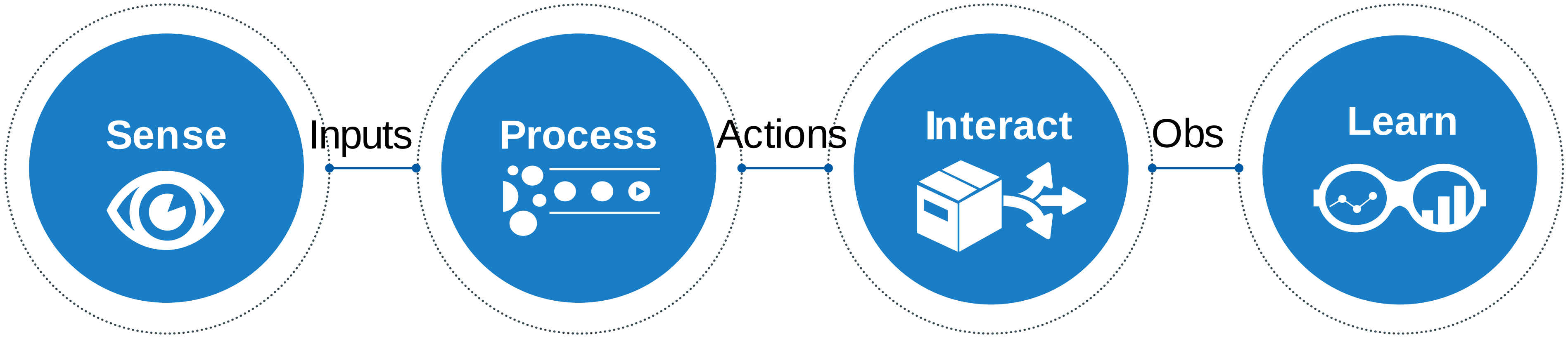
"reasoning"
Execute model to select and post actions

"acting"
Perform the desired actions

Measure, compare, adjust

Learning: could be human methods (manual adjustment) or machine methods (e.g. reinforcement learning), which change the sensing and processing.

ML feedback requires a lot of data that you must record



Record the inputs

Record the action, the expected outcome (tracking metrics and OEC

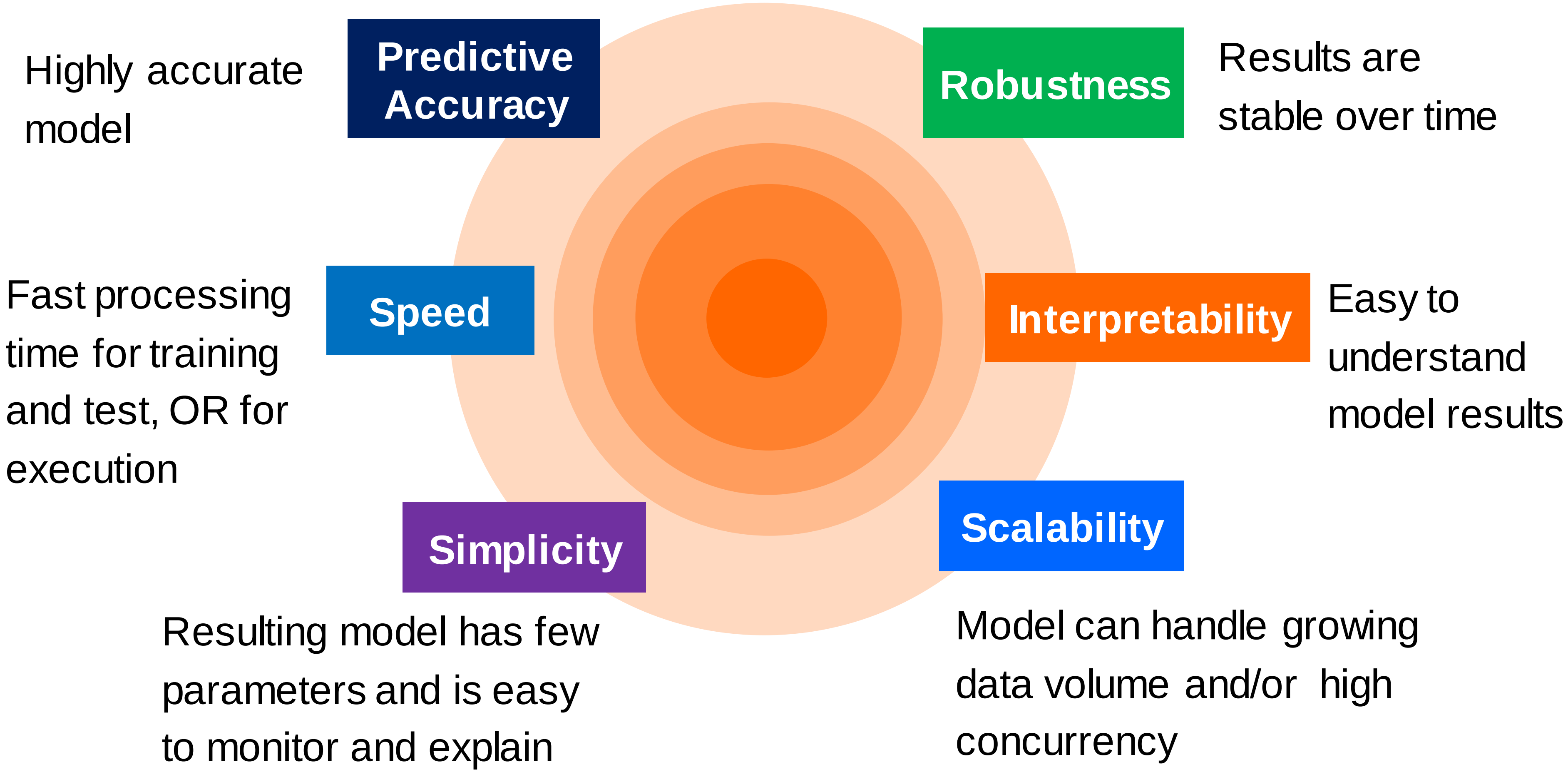
Record the execution. Record the metric deltas

Record the changes

Data volumes explode with all the telemetry:

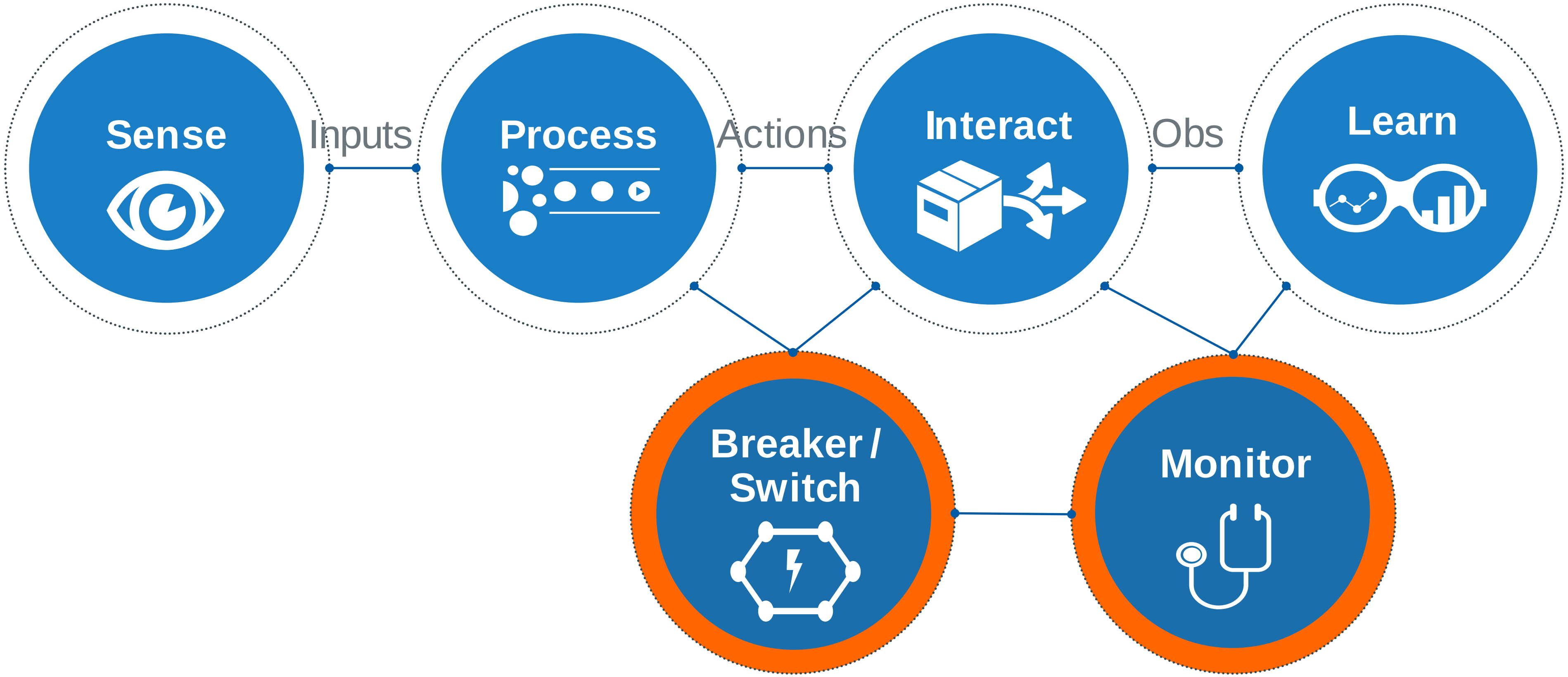
1 ML execution = raw data in, inputs, the action, each metric used (expected values), execution log, metric data (actuals), deltas, model changes, technical resource information

Criteria for models: not just accuracy!



You must track performance in development and production *relative to the metrics that are most important, in addition to the OEC.*

You need to protect against model execution problems



You have to track the actions / executions and their results, including the OEC, in real time, to protect against failures.

This adds monitors and circuit breakers, which need data.

“A murder scene, but with poop” – “We see this a lot”

AI seems smart because you are looking at it too narrowly. The solution to a known problem in isolation can be ruined by one well-placed exception.

A key "smart thing" problem is not having enough context

"In the real world, exceptions are the norm."
– John Gall

What is context to ML? Data. Unknown data is a model failure, a problem.

The smart thing creates the problem it is designed to fix: **that is the definition of an organized crime "racket"**

Until you have better ability to sense, adapt, and learn, you won't have smart things.

Systemic side effects

The lack of context in design affects the people working ***on*** your smart product too.

PhD-level AI practitioners are building 3-D models of poop in virtual worlds.

“Not poop” is the next great frontier for their research.

All of this operations work is about data, including synthetic data creation

Smart design is about agency

When you make a product smart, you are giving it agency in the world.

That agency creates feedback loops: AI changes people, their behavior changes the AI.

You empower a thing, but that may mean you disempower a person.

Therefore:

Design with a focus on the people who are in the range of interaction.

Model decay (or drift) is inevitable

AI and ML applications exist to act or inform people’s actions.

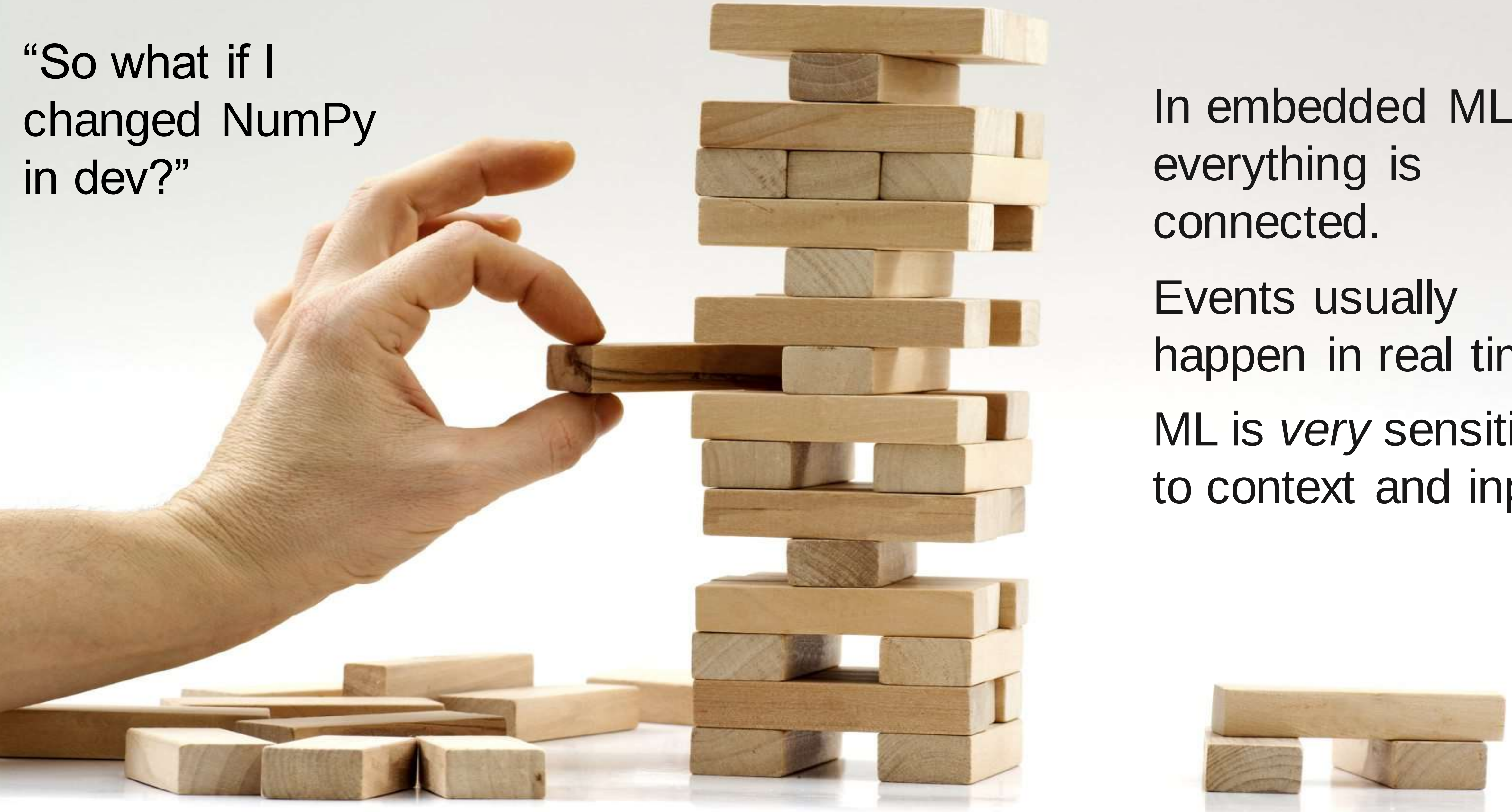
Actions affect the environment, changing it.

Changes to the environment affect new inputs to the model.

It is not the model that is changing. It is reality that is changing while the model remains static.

ML Principle: CACE, Change Anything Change Everything

"So what if I changed NumPy in dev?"

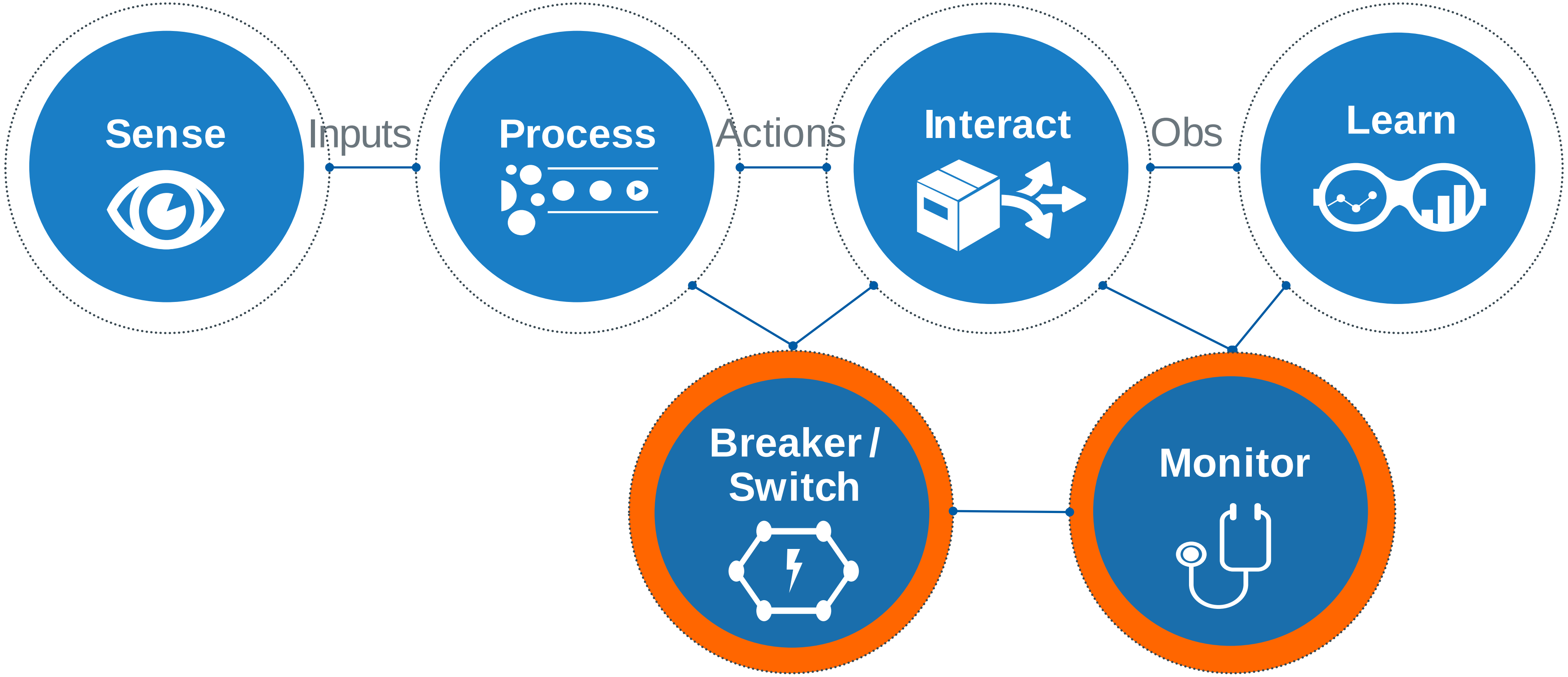


In embedded ML everything is connected.

Events usually happen in real time.

ML is *very* sensitive to context and input.

Not just protection against failures - diagnostics



You need telemetry about the entire environment for monitoring, but you also need it for *diagnostics*.

This means you need to think about *observability.*

“A production ML system is never all green”

Much of the time, the ML app is a distributed system.

Distributed systems are hard.

Monolithic architectures are actually great if you can use them.

ML is not like code: Monitoring in production

Unlike other software, ML has different metrics for “correct”

The metrics are relative and can change over time, in production

Therefore, *production is a part of your testing environment.*

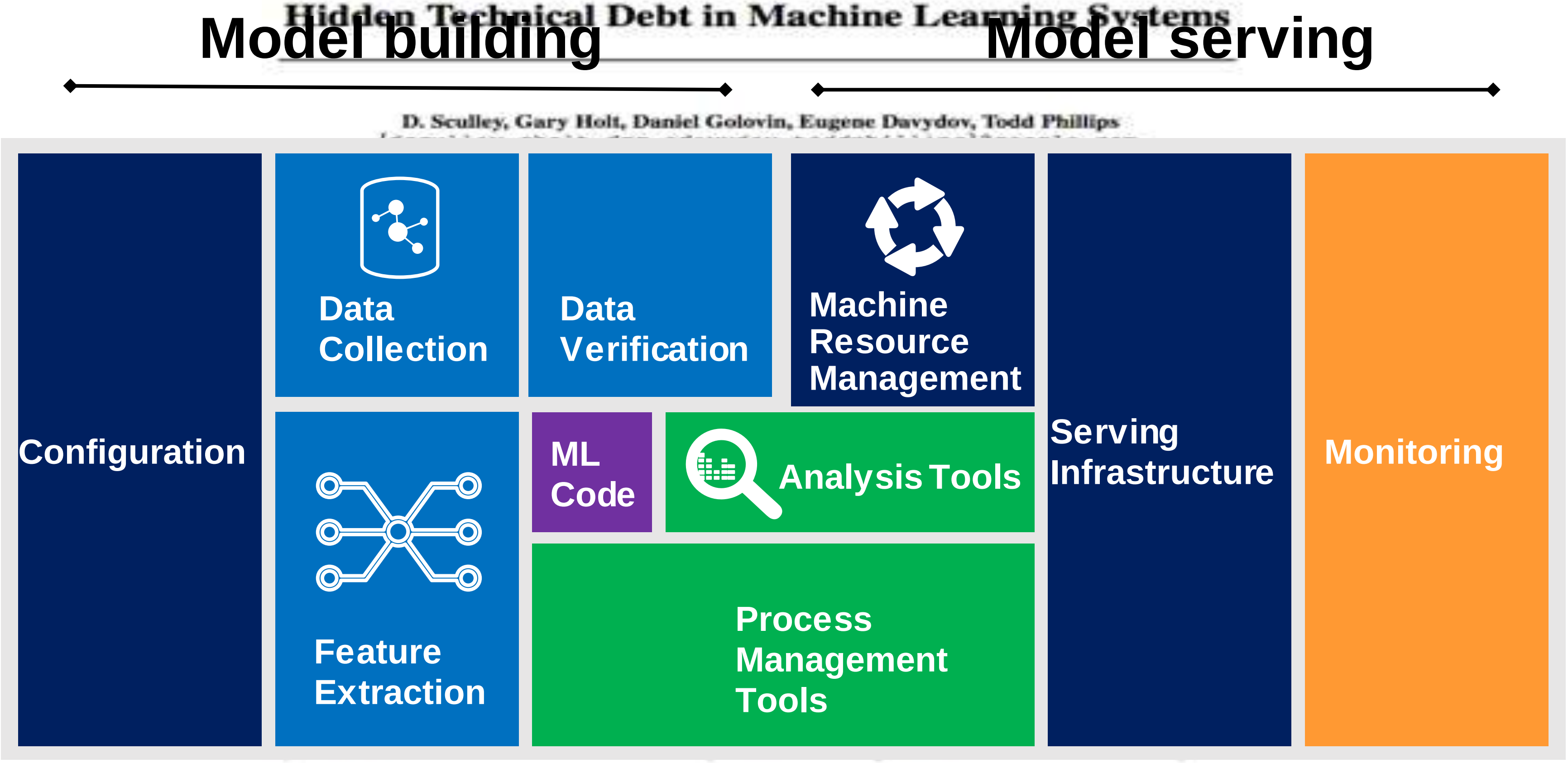
You must monitor performance closely, which is like doing BI on your AI.

“observability”, because a problem *may not be the model but the data, or the infrastructure.*

- **Reduce the time to diagnose, rather than emphasizing the prevention of coding errors**

All models must be monitored, all the time

Machine learning is the smallest part of the environment



Hidden Technical Debt in Machine Learning Systems

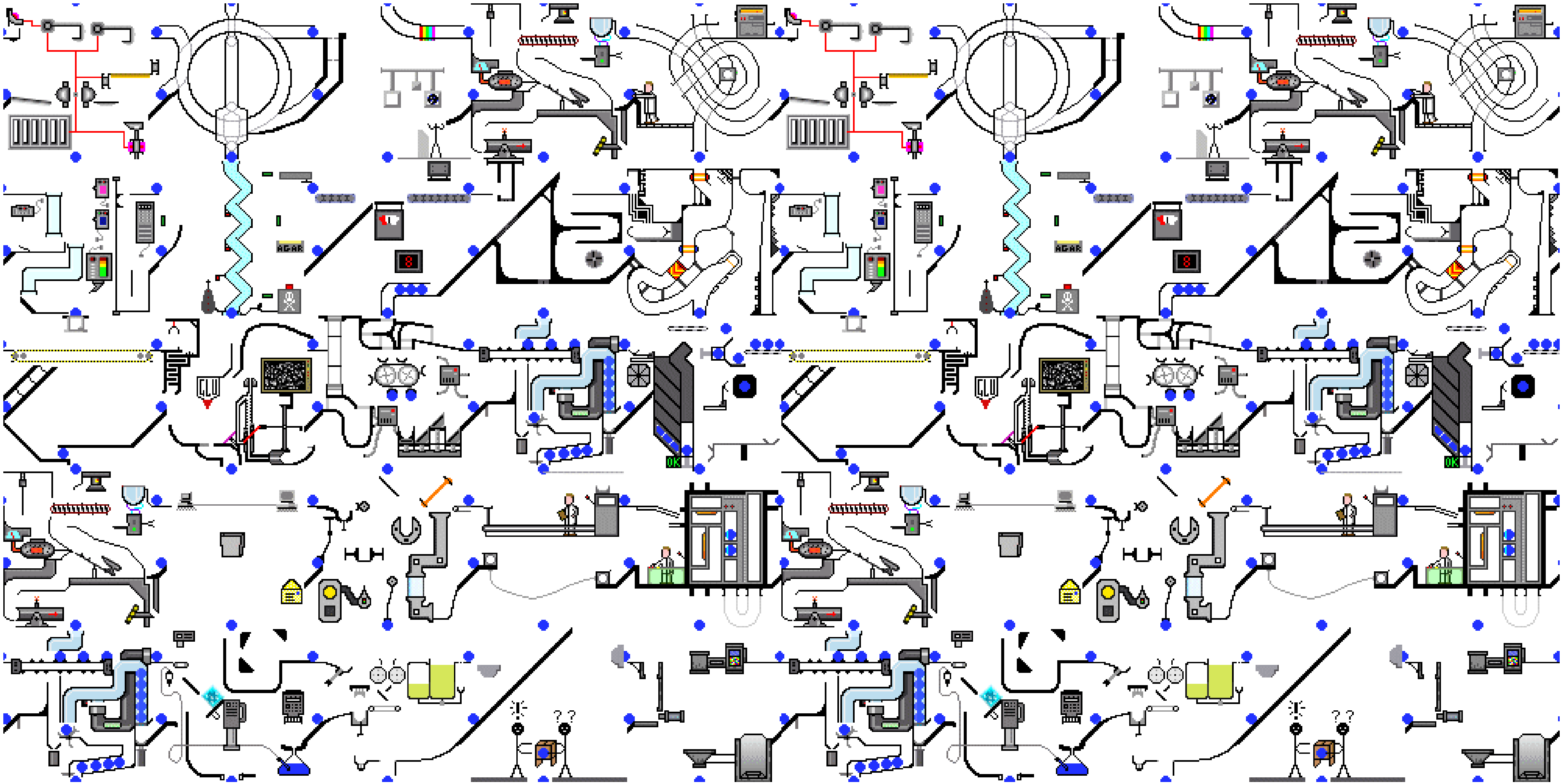
D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips

Static Analysis of Data Dependencies. In traditional code, compilers and build systems perform static analysis of dependency graphs. Tools for static analysis of data dependencies are far less common, but are essential for error checking, tracking down consumers, and enforcing migration and updates. One such tool is the automated feature management system described in [12], which enables data sources and features to be annotated. Automated checks can then be run to ensure that all dependencies have the appropriate annotations, and dependency trees can be fully resolved. This kind of tooling can make migration and deletion much safer in practice.

https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems

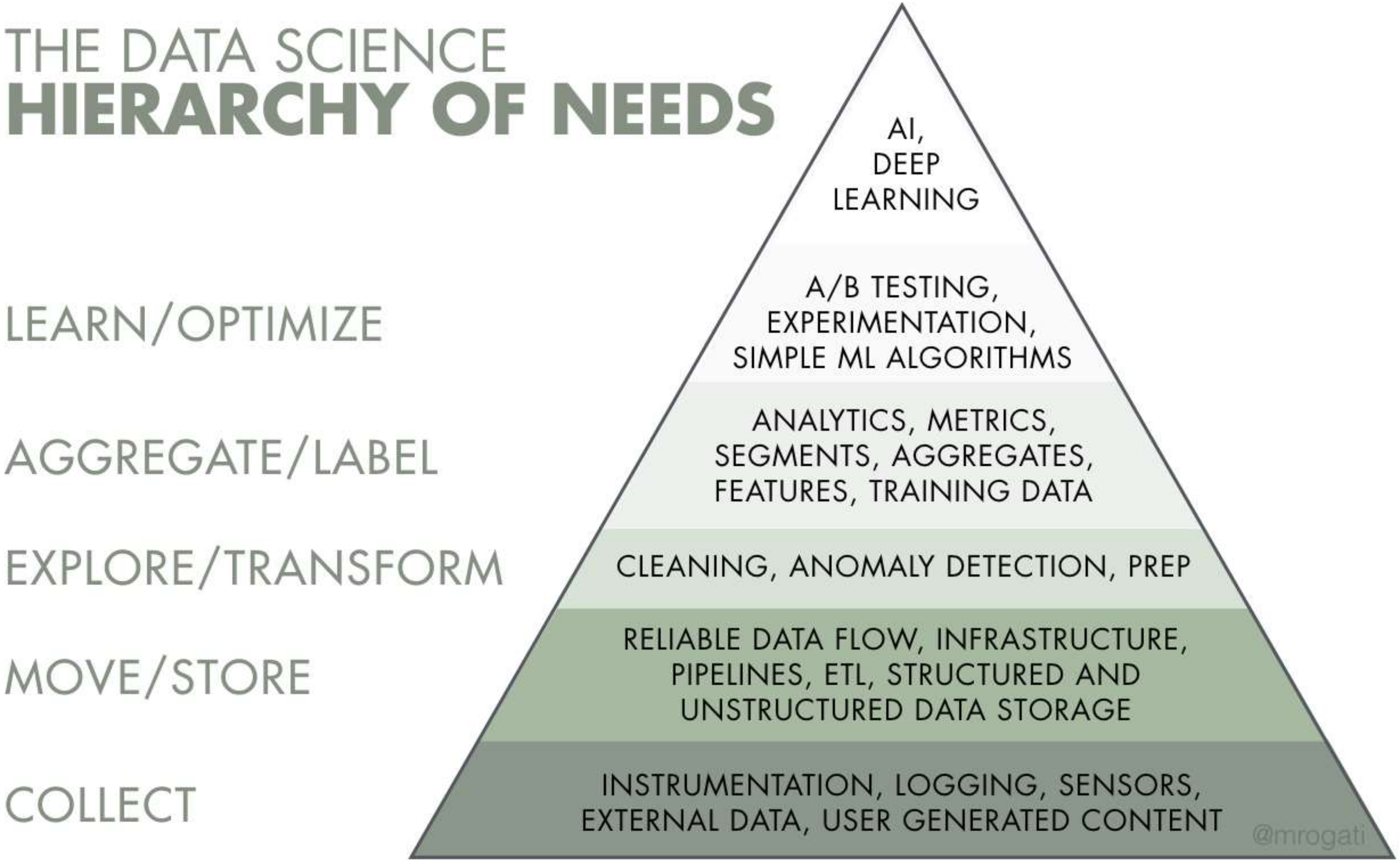
The end result in the data science landscape is complexity

You can explain the ML model, but can you explain this?



Key to operationalizing ML is data governance and data management

(but nobody will believe you)

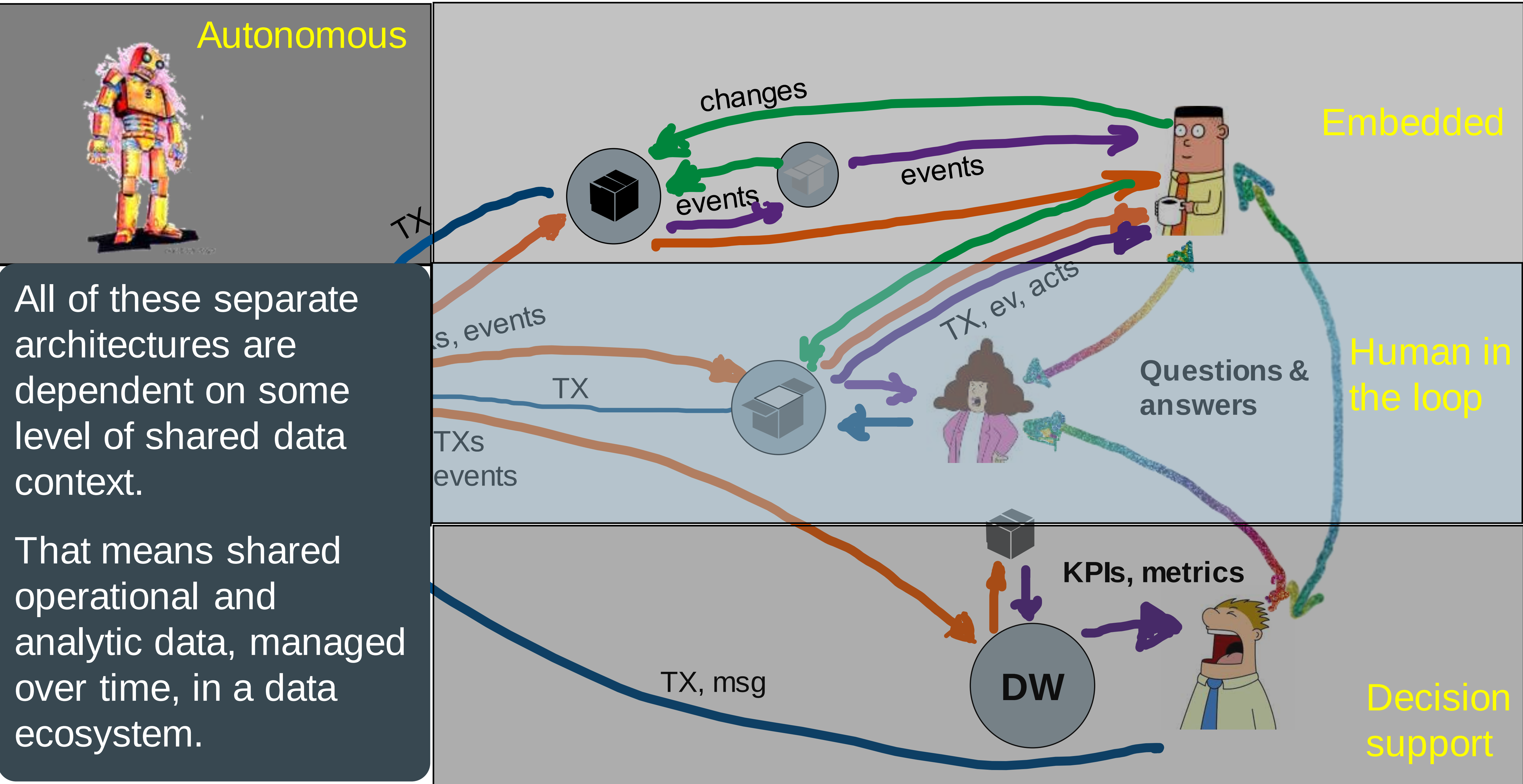


Add: underlying data governance and data management.

There is a lot of work to do before you can get to the interesting parts

https://hackernoon.com/the-ai-hierarchy-of-needs-18f111fcc007

The work in an ML program involves mostly data management

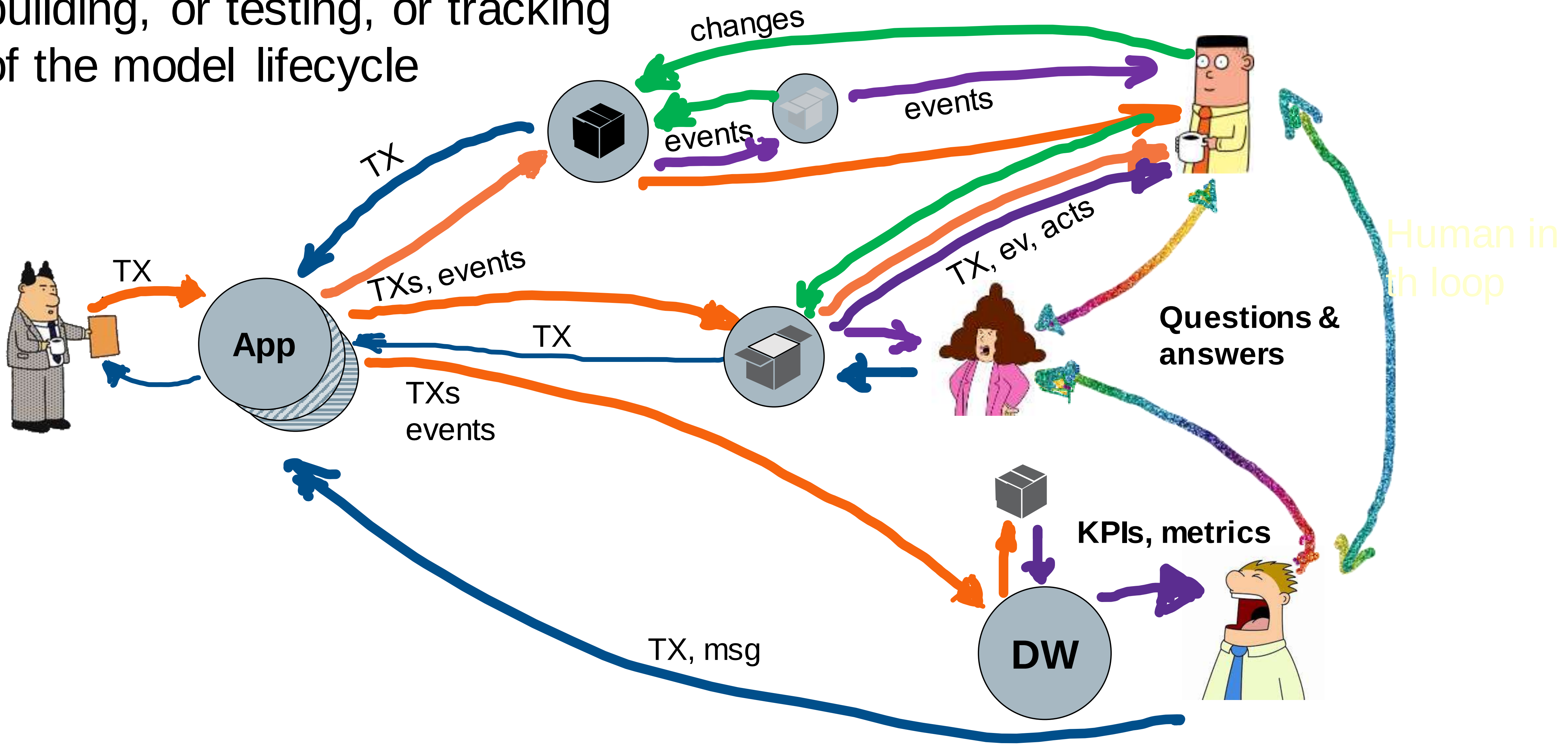


All of these separate architectures are dependent on some level of shared data context.

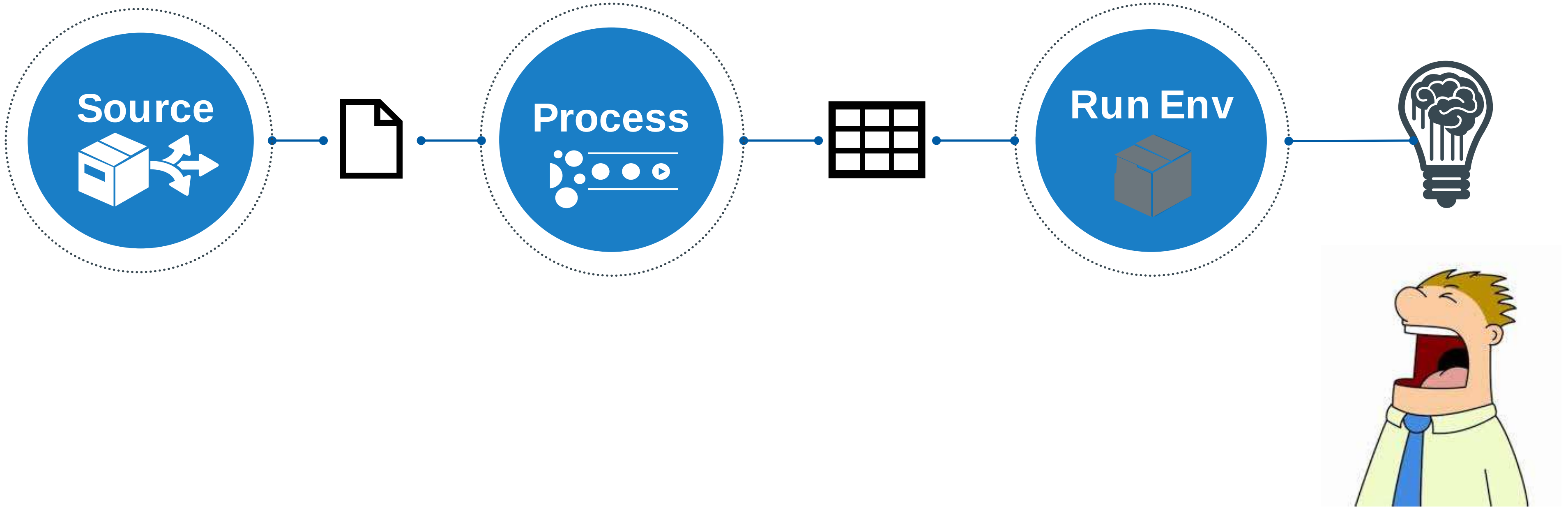
That means shared operational and analytic data, managed over time, in a data ecosystem.

All the arrows indicate data that you need to collect and manage

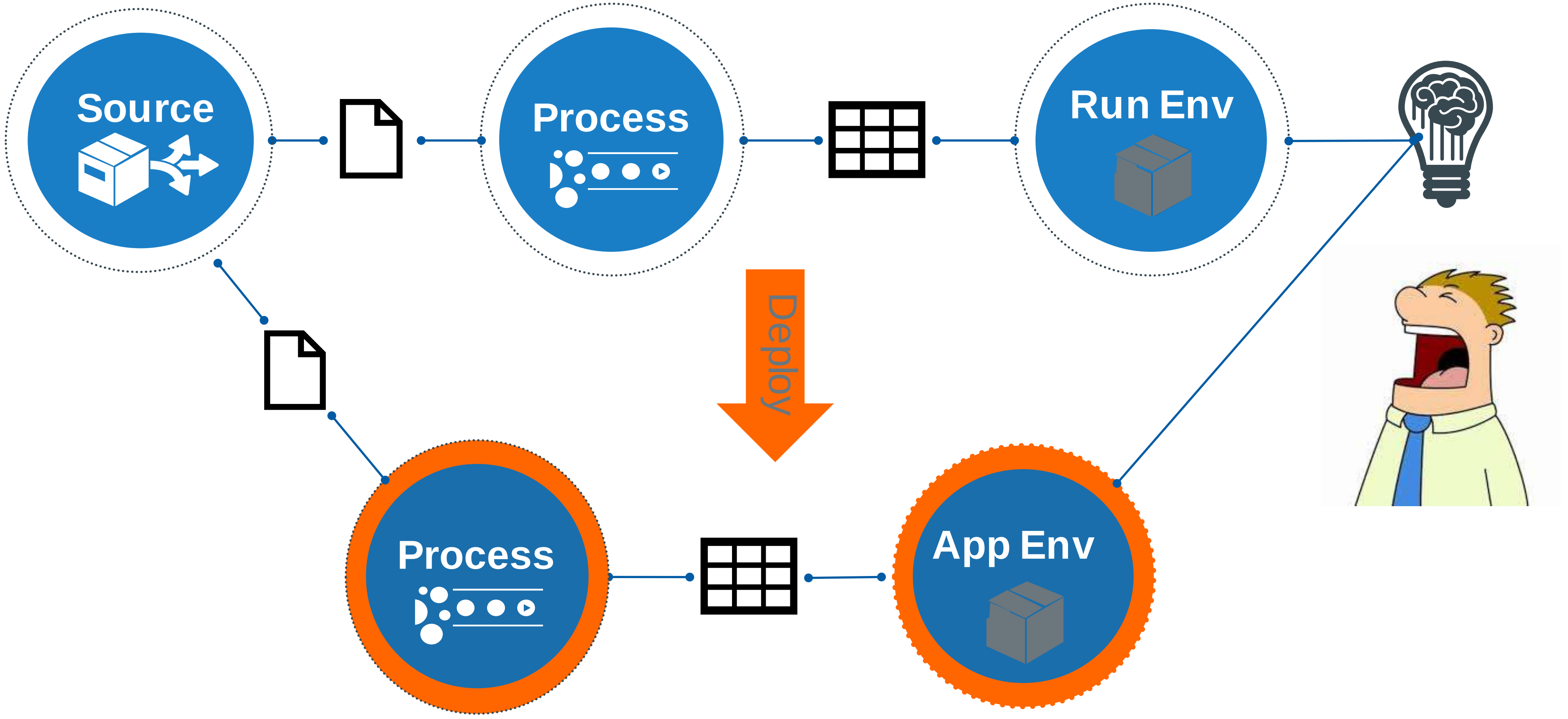
And this doesn't include the building, or testing, or tracking of the model lifecycle



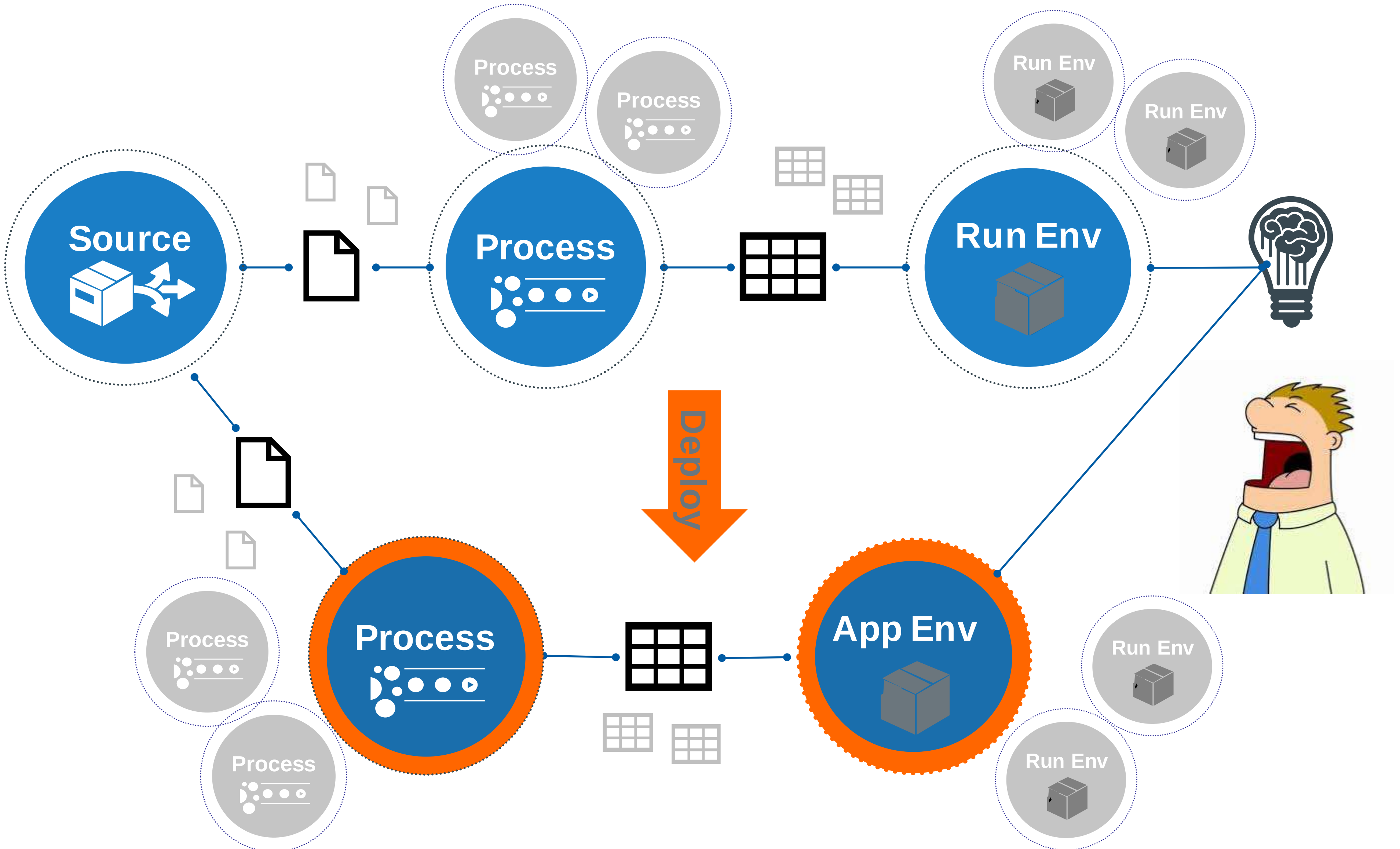
Working backwards from a decision or answer, what do you need to have diagnose or reproduce things?



Fine for a single-run model. What if it's embedded in an application somewhere?



If a model is used over time, you will likely change it. Or sources, or processes will change



Conclusion

Most failures are not technical: start with the core questions

1. Is your goal to address operations for a project, a program, or a product?
2. What type(s) of ML systems will you be building and operating, and how many of them?
3. How will you be approaching the use of those systems?

Then map out workflows and data flows, including *all* people who are affected in any way by the ML application.

Then think about all the different data and how you will manage it.

Culture: Having an experimental mindset

Sometimes you can’t build the thing you want (meet the OEC)

- ML is experimental, you should fail
- Budget to experiment – and fail?
- Data: type, quality, amount
- Technique: theoretical limits, appropriateness
- Feasibility: technical, resources and time

Useful background for online experiments

https://www.researchgate.net/publication/316116834_Online_Controlled_Experiments_and_AB_Testing

https://ai.stanford.edu/~ronnyk/2007GuideControlledExperiments.pdf

Managing the code without managing the data?

Most emphasis in the industry is on code and code artifacts:

- Model repositories
- Model management
- Pipeline frameworks
- Packaging
- Versioning
- Tools

Why? Because vendors want to sell you products for the problem they helped create.

What do the experts say?

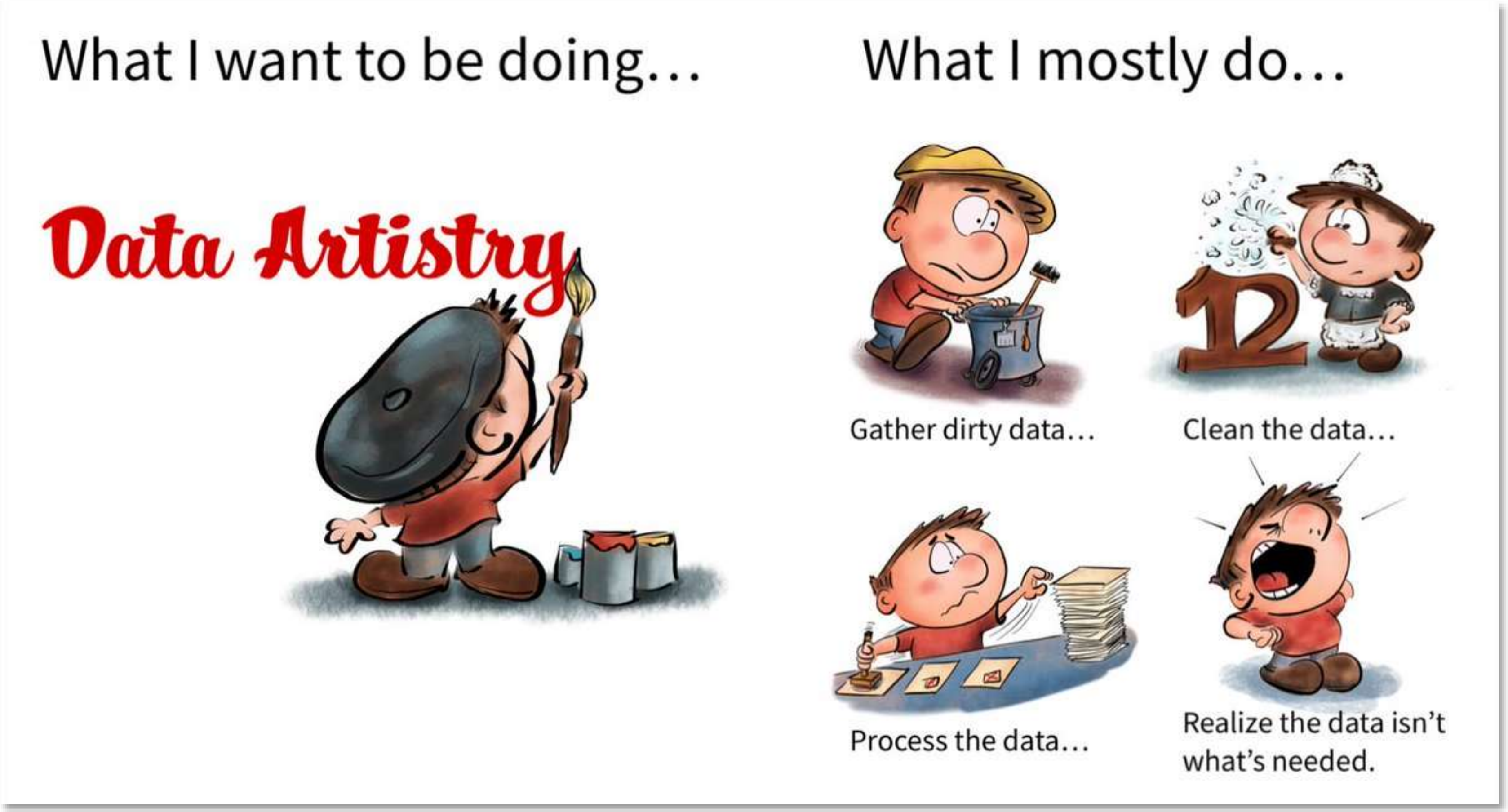
TIDY DATA: Hadley Wickham makes the case for Tidy data sets, that have specific structure, are easy to work with, that free analysts from mundane data manipulation chores – there's no need to start from scratch and reinvent new methods for data cleaning

Source: Tidy Data by Hadley Wickham, Journal of Statistical Software, Vol 59, issue 10 (2014)
https://vita.had.co.nz/papers/tidy-data.html

"Eliminate the time spent on data prep" – Wrong

The work you do on the data is what makes it valuable.

You can't eliminate the prep work without eliminating good models. Instead, optimize workflows where most of the time is spent.



Your AI only knows about what is visible in the data

For an AI, data is the world.

What happens if communications fail? The network fails, or a sensor fails, or security fails? A lack of data.

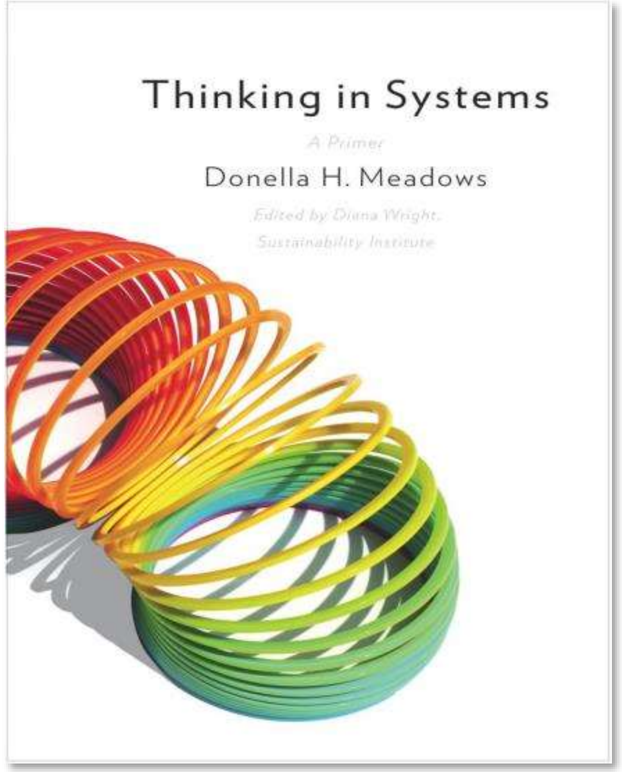
Does your smart thing fail gracefully, or like someone who doesn’t care?

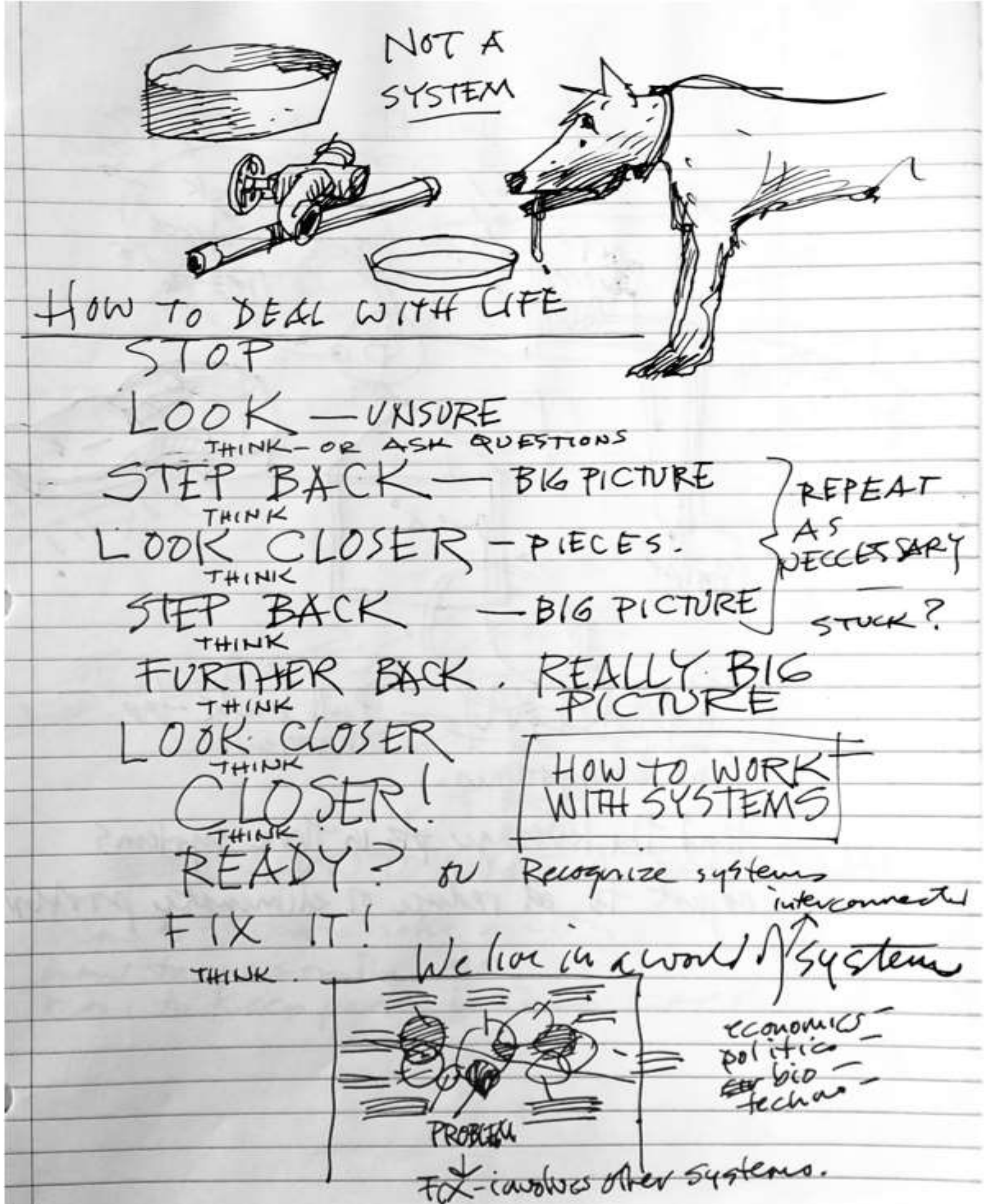
How can it harm the user?

The operational design extends beyond the code, and the data.

Every negative effect of ML is a direct result of human acts in the context of the organization that created it.

Model the entire system, including yourself in it





References

Other helpful general references

Systemantics: How Systems Work and Especially How They Fail, aka The Systems Bible, John Gall 1978, 2003, https://en.wikipedia.org/wiki/Systemantics

Thinking in Systems, Donella Meadows, https://donellameadows.org/systems-thinking-book-sale/

Exploring social network effects on popularity biases in recommender systems, http://ir.ii.uam.es/pubs/rsweb2014.pdf

Ten Challenges for Making Automation a “Team Player” in Joint Human-Agent Activity, https://ieeexplore.ieee.org/document/1363742

The story of socio-technical design: reflections on its successes, failures and potential, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.455.490&rep=rep1&type=pdf

About the Presenter

Mark spent most of the past 25 years working in the analytics field, starting in AI at the University of Pittsburgh and autonomous robotics at Carnegie Mellon University before moving into technology management. Today he is a Fellow in the Technology & Innovation Office at Teradata. Previously, he was president of Third Nature, an advisory firm focused on services for analytics and technology strategy, and product design.

Mark is an award-winning author, architect and CTO who has received awards for his work from the American Productivity & Quality Center, Smithsonian Institute, and industry associations. He is an international speaker, and chairs several conferences and program committees. You can find him on LinkedIn at https://www.linkedin.com/in/markmadsen

teradata.

End