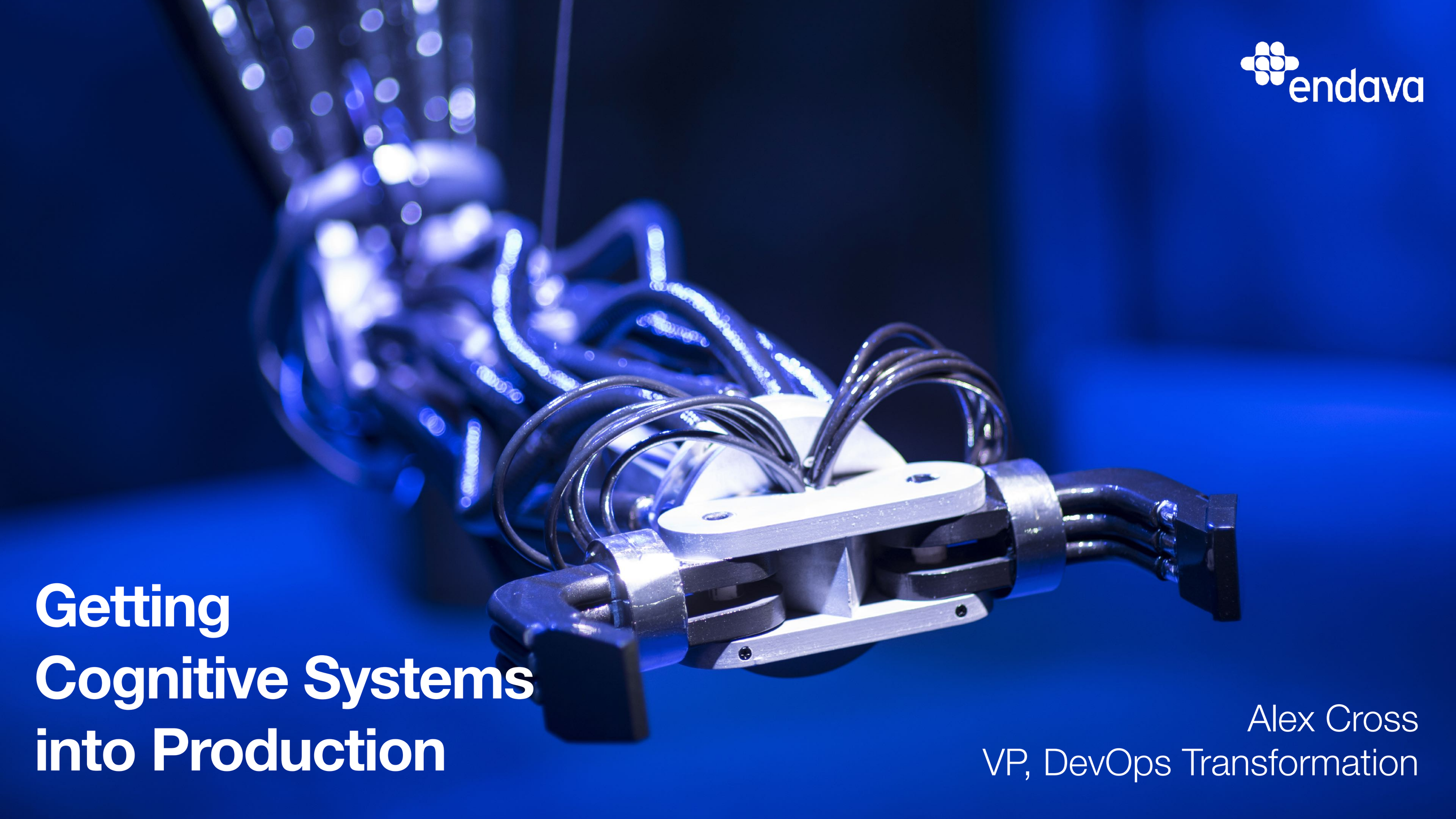

endava



Getting Cognitive Systems into Production

Alex Cross
VP, DevOps Transformation

I

CONTEXT

BUSINESSES WANT TO MOVE BEYOND PURE ANALYTICS USE CASES

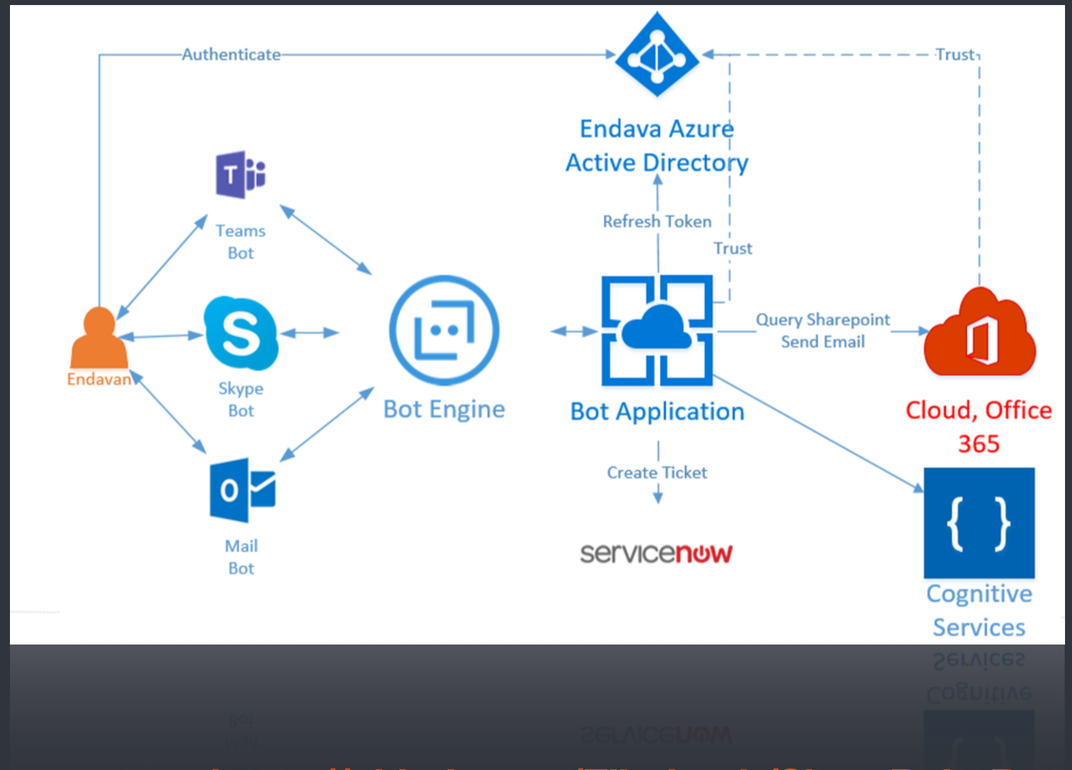
Analytics is great, and of course, ML extends this
But insights, graphs and reports don’t have to be the end product.

Cognitive Computing uses ML components in a larger system or automated process.

GREAT COGNITIVE USE CASES ARE BEING EXPLORED *RIGHT NOW…*

Systems that:

- Replace static logo images across 1000s of hours of video content
- Add subtitles to video dialogue on the fly, in a different language
- Give customer service agents the expert answers they need, moment by moment, by following along and understanding the call
- Route scattered police forces to areas of possible crime based on crime stats and global twitter activity
- Classify, rank, resolve and route IT helpdesk queries as they come in



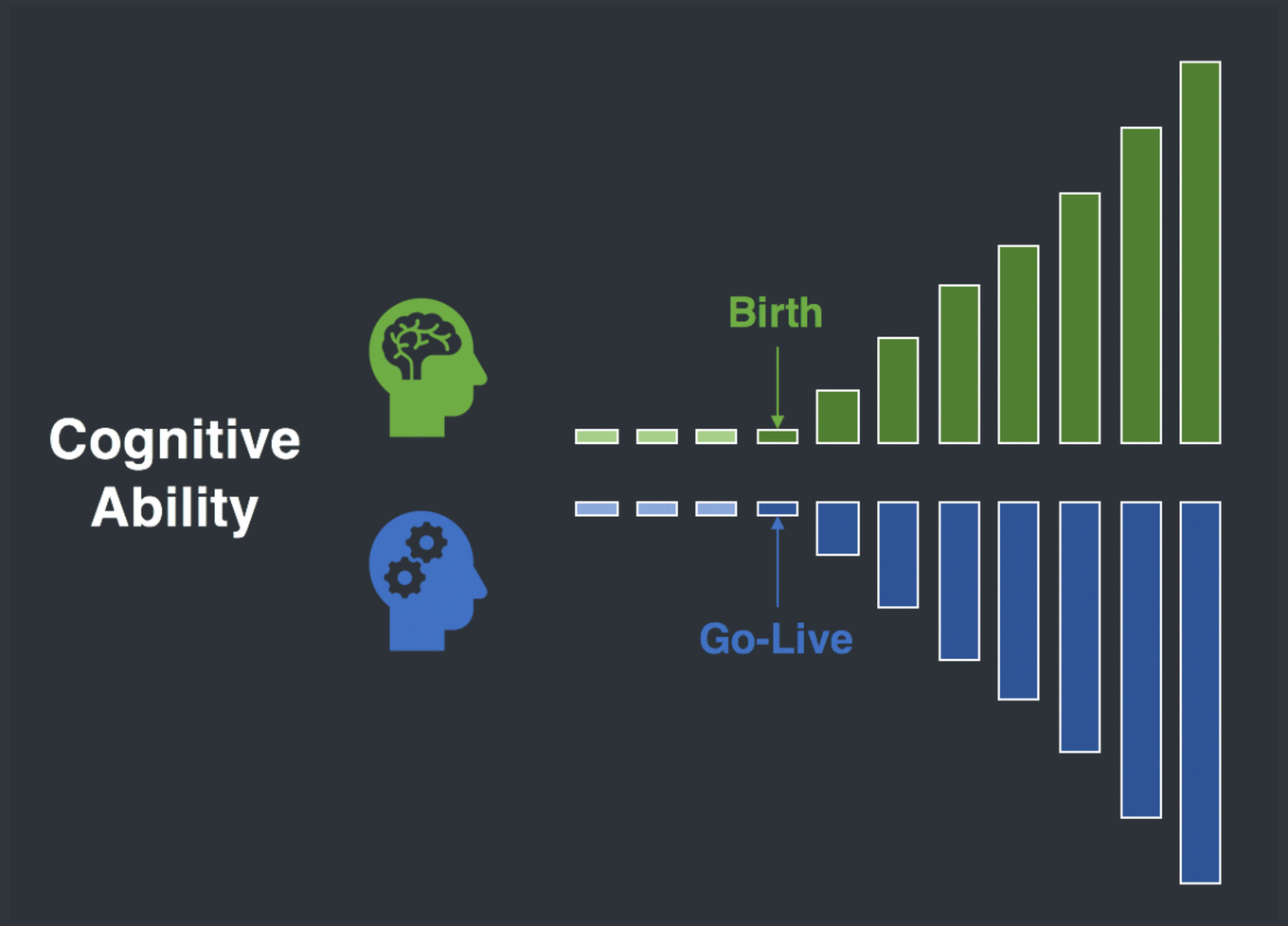
https://github.com/Ellerbach/SharePointBot

https://github.com/karolzak/Support-Tickets-Classification

BUT MANY WILL NEVER GO LIVE.

II

PRODUCTIONISATION PROBLEMS



BUSINESSES DO NOT EXPECT SYSTEMS

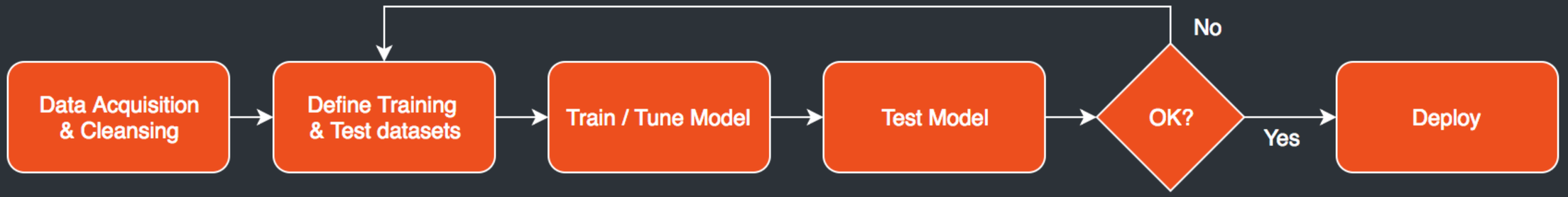
TO PERFORM POORLY ON LAUNCH DAY.

INTEGRATING ML INTO LIVE SYSTEMS & PROCESSES IS HARD.

Cognitive systems are:

- **Hard for business stakeholders to understand**
- **Hard to maintain without specialist ML skillsets**
- **Hard to maintain… in general.**

SIMPLIFIED SUPERVISED LEARNING LIFECYCLE



- This process repeats on a continuous basis, but mostly training / testing is in a non-live environment
- Parts of the process can be automated, but some data science expertise is required somewhere
- The system won’t improve itself

BUSINESS WORRIES FOR COGNITIVE

- How can we specify a useful backlog of ‘features’ for machine learning?
- How can we estimate how long development / experimentation will take?

It’s hard for business folks to understand that…

- The system will make mistakes that cannot be easily debugged
- The re-training process typically involves regressing stuff that used to work OK
- Training and improvement is broadly boundless and ongoing
- You can’t just correct all the system’s mistakes and add all the right answers to the training set

And that’s before we get onto…

- Nuances of unsupervised and reinforcement learning lifecycles
- Human disagreement over expected behaviour

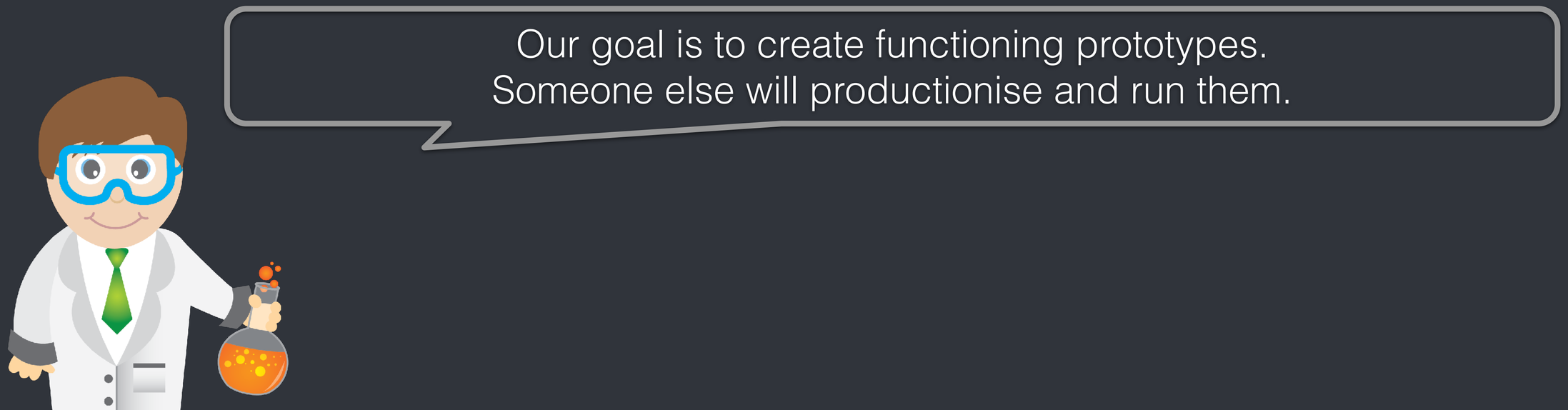
WHY DO WE NEED ML SPECIALISTS IN THE ‘LIVE’ TEAM?

- Hard to run something if you don’t understand how it works or what it does (obviously)
- Need to respond quickly to issues and new requirements on the live system
- Need to continuously improve performance & tune models
- …all without **overfitting**, regressing or breaking the model.

CONTINUOUS IMPROVEMENT OF A LIVE COGNITIVE SYSTEM REQUIRES A UNIQUE AND CROSS-FUNCTIONAL ML SKILLSET.

WHY IS MAINTENANCE STILL DIFFICULT WITH EMBEDDED SPECIALISTS?

- Data scientists and software engineers have different backgrounds, mindsets, specialisations, skills, …and often conflicting priorities.
- Focus on iterating really fast to test, adjust, optimize and retest until hypothesis sufficiently dis/confirmed
- …so writing production grade code to high engineering standards is logically low down on the DS agenda. Prototypes are not meant to be production-grade, after all.



III

GETTING TO PROD

THE INDUSTRY IS CLOSING THE GAP BETWEEN THE DS LAB & REAL-WORLD DEPLOYMENTS

SaaS products continue to emerge to streamline the process of getting models live, at all levels of abstraction

- Complete cloud-based ML workbench environments
- Experiment, ML training and data versioning pipelines
- ‘ML as a service’
- Black box ML APIs

…and 100s of other platforms & vendors

All of these products are grasping at the problem of how to productionise Machine Learning.

BUT LIFECYCLE TOOLS RARELY SOLVE PEOPLE & PROCESS PROBLEMS

(LIKE THE PROBLEMS WE'RE EXAMINING NOW)

THE DEVOPS ORIGIN STORY

Once upon a time, IT Development teams used to write code and throw it at IT Operations teams to deploy and run.

- The **developers** were motivated to build cool things really fast
- The **operators** were motivated to keep things running and stable for as long as possible

The two groups did not get on very well.

- Their **priorities often conflicted** – stability vs. rapid innovation
- They had **different backgrounds**, mindsets, specialisations and skills

Then one day they discovered **DevOps**, which was a way they could work together and share their priorities…

LESSONS FROM DEVOPS

Everyone agreed that building cool things fast was important,
but it was only sustainable by:

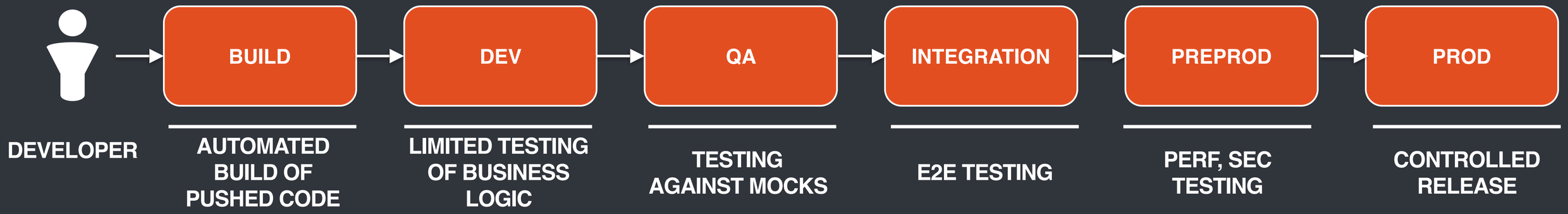
- Committing to high code quality and continuous improvement
- Building against production-like environments
- Extensive automated testing and TDD/BDD
- Sophisticated monitoring, alerting and self-healing capabilities,
- Resilience, security and availability built in up front

They were able to agree on these shared priorities because they worked closely together towards a shared goal.

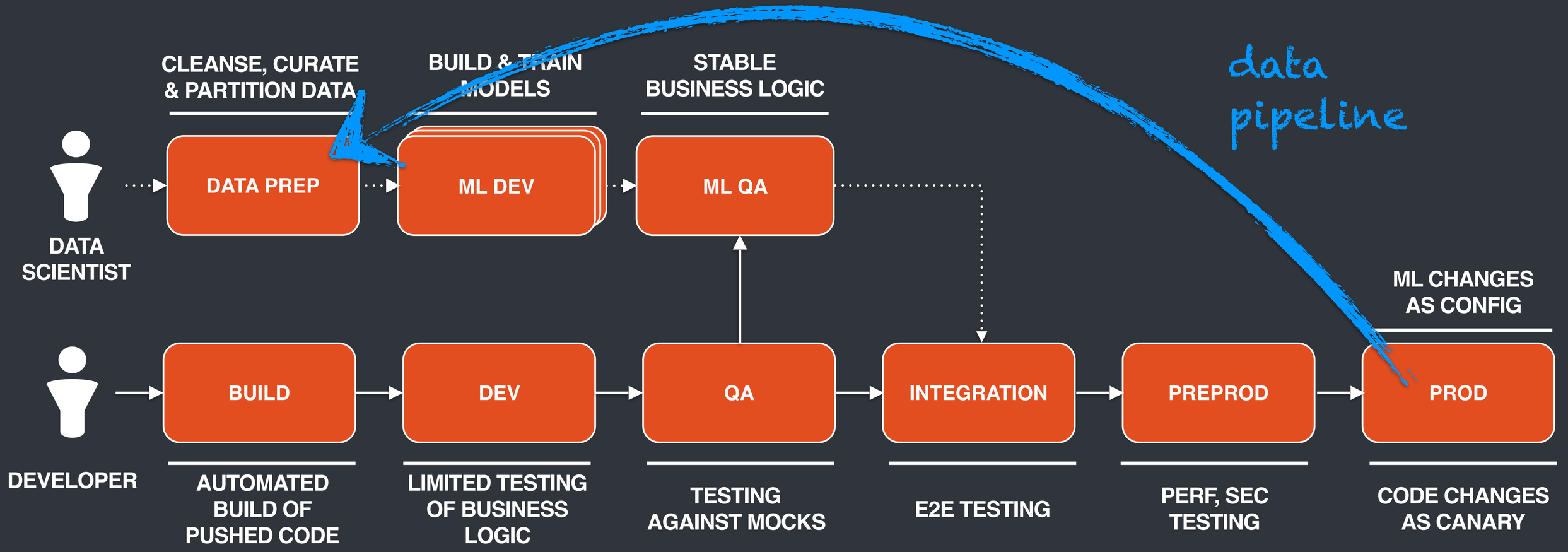
(And they all lived happily ever after etc. etc.)

THIS IS ALSO A RECIPE FOR GETTING COGNITIVE SYSTEMS INTO PRODUCTION.

TYPICAL CD PIPELINE STRUCTURE



CD PIPELINES FOR COGNITIVE SYSTEMS



INCREASING MAINTAINABILITY BY ADOPTING CI/CD PRACTICES

CODE FOR PRODUCTION	Notebooks are great, but at some point your code needs to become a reusable library
PROD-LIKE ENVIRONMENTS	Write your models into Docker containers: write once, run anywhere
IMPLEMENT UNIT TESTS	Automated testing of small sections of code helps prevent unintended variables
STRONG VERSION CONTROL	Library code - but also notebooks, experiments, datasets, tests, *pipelines*…
CODE REVIEW	Helps the team share expertise and define a common standard
AUTOMATED DATA PREP	Configure an automated data pipeline to help provision, cleanse and partition your datasets
FREQUENT E2E TESTING	Expand your horizons to cover a sample of full-stack use cases, and test against them frequently.
PLAN FOR MONITORING	Validating the behaviour of the live system will be a statistical exercise in itself using ‘secondary’ data sources, and can feed into the experimental process

WHAT ABOUT HELPING THE BUSINESS?

E2E TESTING IS KEY BUT REQUIRES INNOVATION

- Keeping track of desired behaviour should be high on any Product Owner's agenda
- But exhaustively specifying behaviour is counter-productive, unmanageable and unrealistic
- Focus on key scenarios and write black box test scripts
- These test expected behaviour from user's perspective
- This helps the business get to grips with what you are building and articulate acceptance criteria

(…though you may need a custom testing setup.)

```
- customer: "Hi I need to renew my motor polic
- bot: "Hello ${customerName}, thanks for cont
- bot: “Before we start, please can you confir
holder?"
- customer: "Yes"
- bot: "Great, thank you."
- bot: "Just to confirm, is this the policy yo
- customer: "Yes that’s right"
- bot: ”Thank you. Let me just check your renew
- bot: ”Ok, the premium is £${renewalPremium}"
- customer: ”Ok.”
- bot: ”Please follow this link to our payment
                ${transactionDeeplink}”
```

TAKE THE SHORTEST PATH TO VALUE.

Quickly delivering a valuable PoC increases its chances of survival in the wild.

DO

- Pick a well-bounded problem domain where you know data will be available
- Stay close to the end users and operators and think carefully about UX
- Consider pre-built, SaaS and open source components (models, integrations, everything) to limit your toil
- Get something into the hands of your stakeholders ASAP, and focus their attention on the ML aspects
- Start planning how you will ramp up volumes of real data from day 1 of working on the PoC

DON'T

- Confuse production-readiness with perfection
- Try and design the thing by committee
- Expect to pass launch without any business change

IV

CONCLUSION

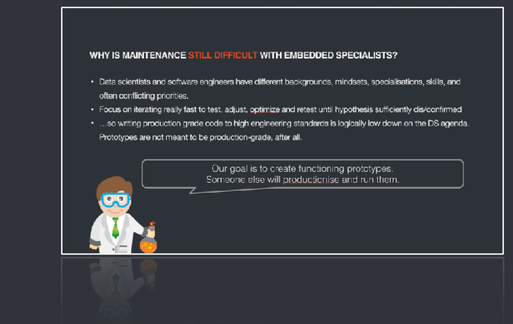
REMEMBER.

- **Cognitive systems are hard for business stakeholders to understand**

… but much easier with reference to **E2E tests** and an **early prototype**

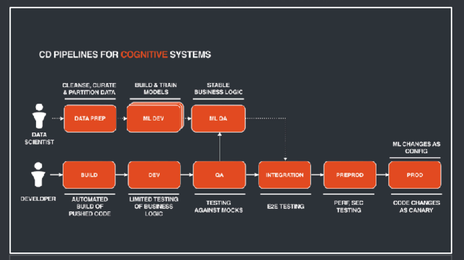
- **Cognitive systems are hard to maintain without specialist ML skillsets**

… so scientists and engineers need a **shared goal** & **common priorities**



- **Cognitive systems are hard to maintain anyway**

… but **DevOps engineering techniques** will ease the Path to Production.



THANK YOU

Alex Cross
Vice President, DevOps Transformation
alex.cross@endava.com