The Long Journey of Legacy Modernization


nick-tune.me


@nick-tune.me


hachyderm.io/@nick_tune


/in/nick-tune

Techies: We understand that your current workflow is very frustrating.

You have to use three legacy systems to get simple tasks done and you have lots of work-arounds and spreadsheets.

We're here to help. We're going to build 1 modern system that does everything with a great UX.

The users: No, don't do that! We do not want that!

The users: Because we used to have two tools. Then we were promised one tool to replace them all and now we have three!

We do not want a fourth!

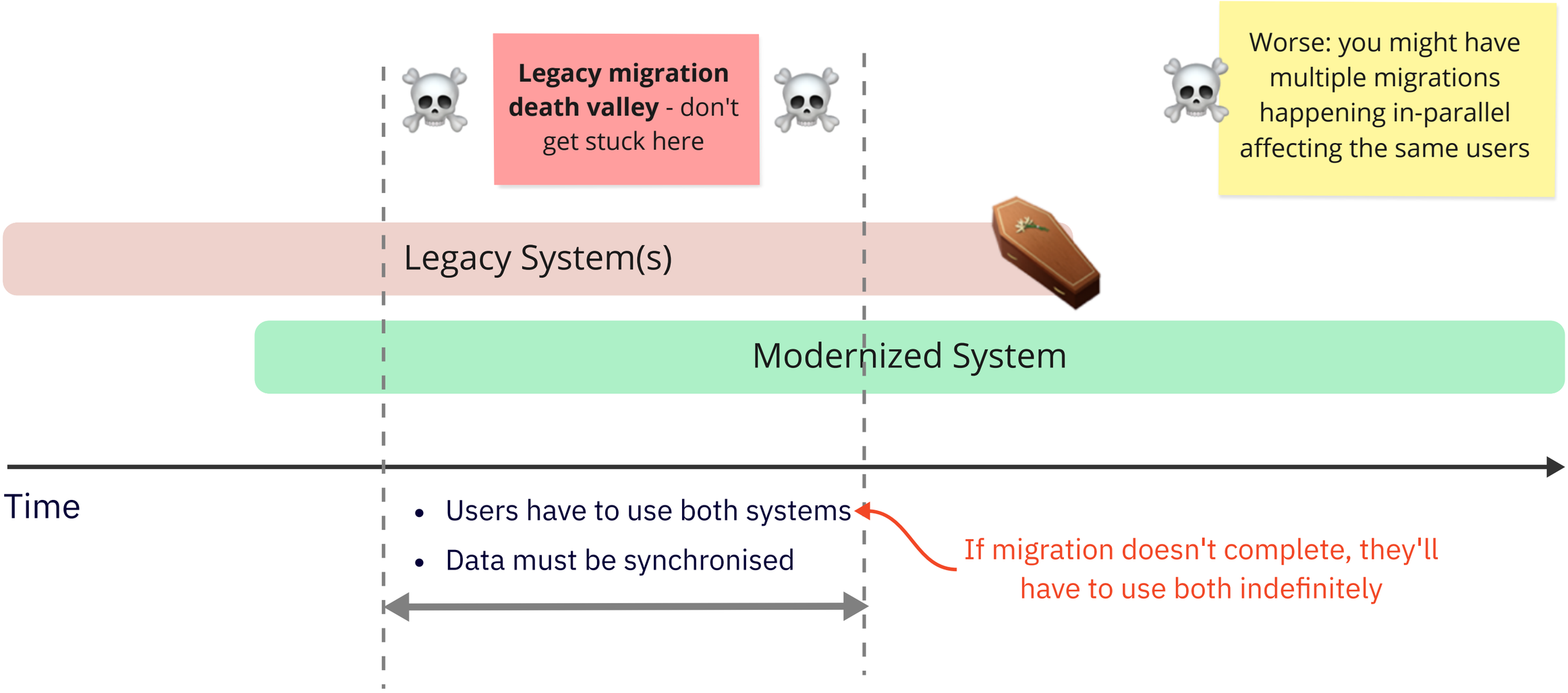
I prefer to not do this migration unless we commit 100% to finishing. Let's stick with what we have.

We have a history of unfinished migrations and we have a lot of extra complexity that takes a lot of effort to maintain.

-- Senior developer

Death Valley

Legacy migration death valley: New and old running in parallel... forever



My friend James ignoring the warning signs in death valley

How to avoid legacy migration death valley... and achieve transformational business growth opportunities?



Challenge 1:
The modernization skills gap

We were falling from the microservices tree, hitting every branch on the way down. Instead of enabling us to move faster, the small team found themselves mired in exploding complexity.

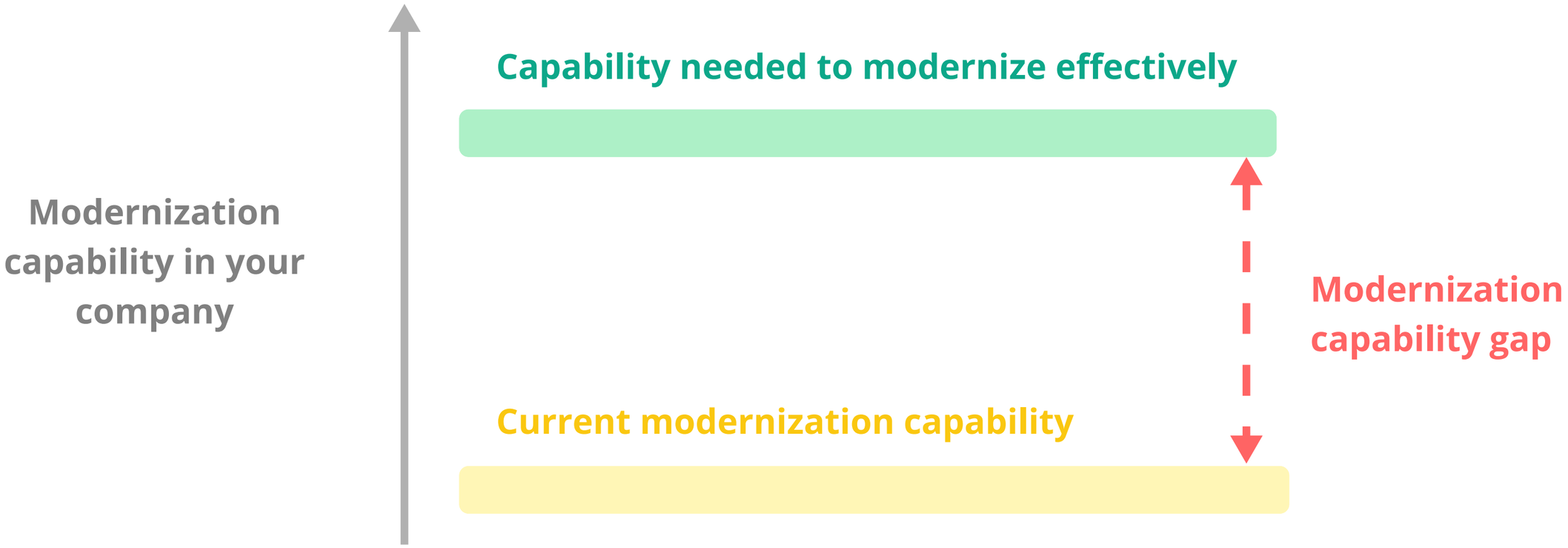
As our velocity plummeted, our defect rate exploded.

https://segment.com/blog/goodbye-microservices/

You can't just ask teams to start modernizing... it's a different mindset and skillset

You risk recreating all the old problems (or even worse new ones)

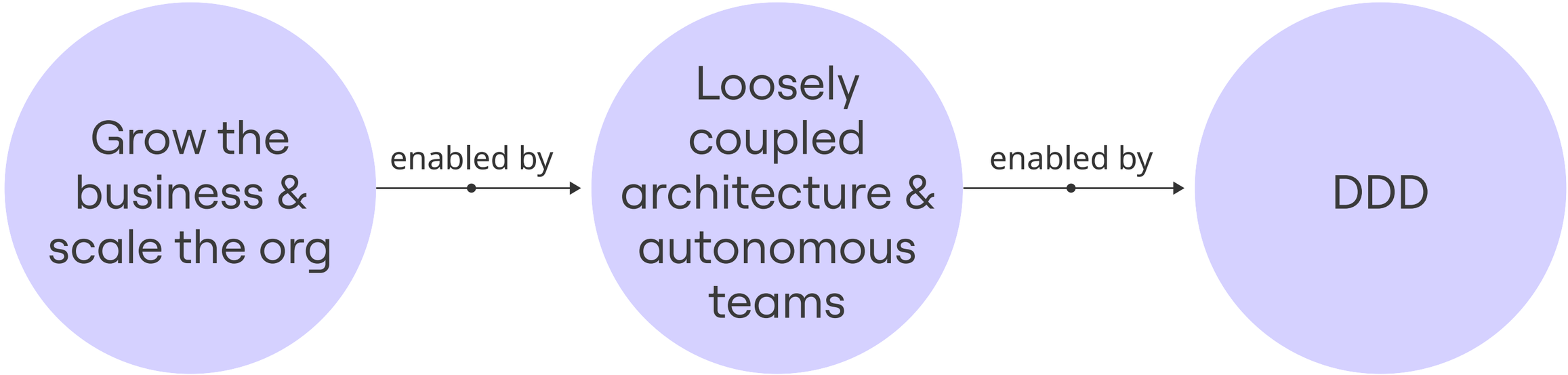
The modernization capability gap



Industry Example

Adopting, embedding and mastering DDD at PayFit

Why DDD (domain-driven design) ?



How to truly upskill and embed DDD?

Just send the tech leads on a 2-day DDD training

Establish regular and ongoing learning and support activities

Krisztina Hirth, Staff Architect

Organizer of DDD Open Hours

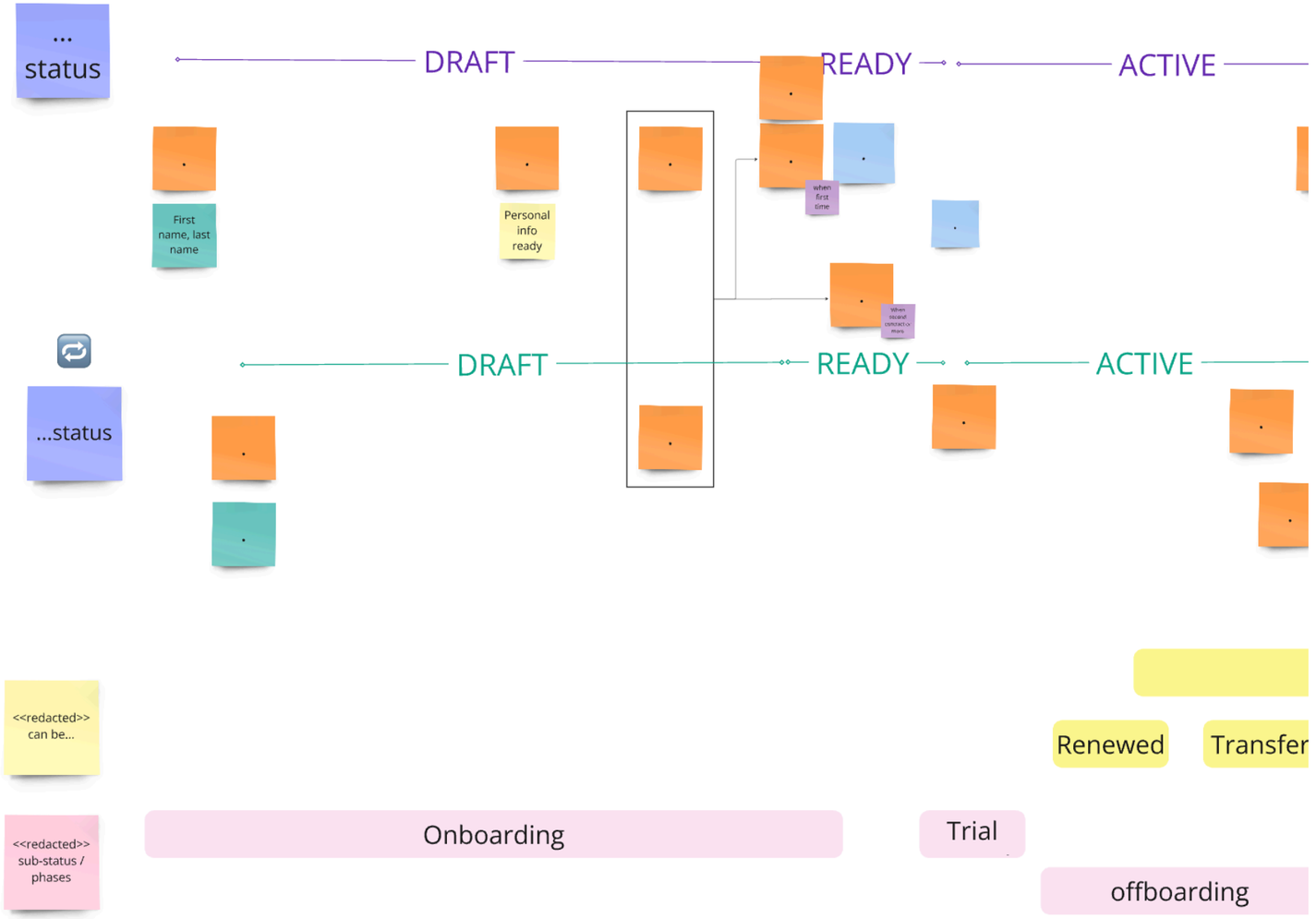
PayFit DDD Open Hours – every Monday

- Any team can bring any topic they want help with - general concepts or applied to real projects
- Anybody can attend, not just developers
- Krisztina and other DDD experts are there to support and facilitate sessions
- Organizers seek out opportunities in the organization for open hours topics
- Constantly remind people open hours is available and encourage them to use it

DDD Open Hours Example: Mapping out current + future entity lifecycle and defining key events

Session Objective
Define the target lifecycle for an entity and public events that will be produced by Team A.

Team B requires the events to transform their architecture - moving to EDA & DDD etc.



What else we do

Hands-on tactical DDD sessions

Internal DDD blog

Staff Community

RFCs & guidelines

Engineering strategy

DDD community + slack channel

Embedded in teams, pairing with developers

Create a safe space to let people make mistakes and learn

Challenge 2:
Making significant decisions

Indecision Consequence:
Short-term workarounds and hacks that add complexity and delay modernization

"We are waiting to be told which team will own *{thing}*. Until then we will build a tactical solution on our side."

Indecision Consequence:
Modernization doesn't get done

"We are still waiting for clear product strategy decisions.

Let's keep building in the legacy until we know this is worth modernizing."

Industry Example

Defining a target model so affected teams can progress confidently

Context

As architecture transformation goes deeper and deeper across the whole system with many teams involved, team ownership and architecture design for some parts of the system and some problems are unclear.

If decisions are not made, teams will soon be blocked from modernizing or forced into workarounds.

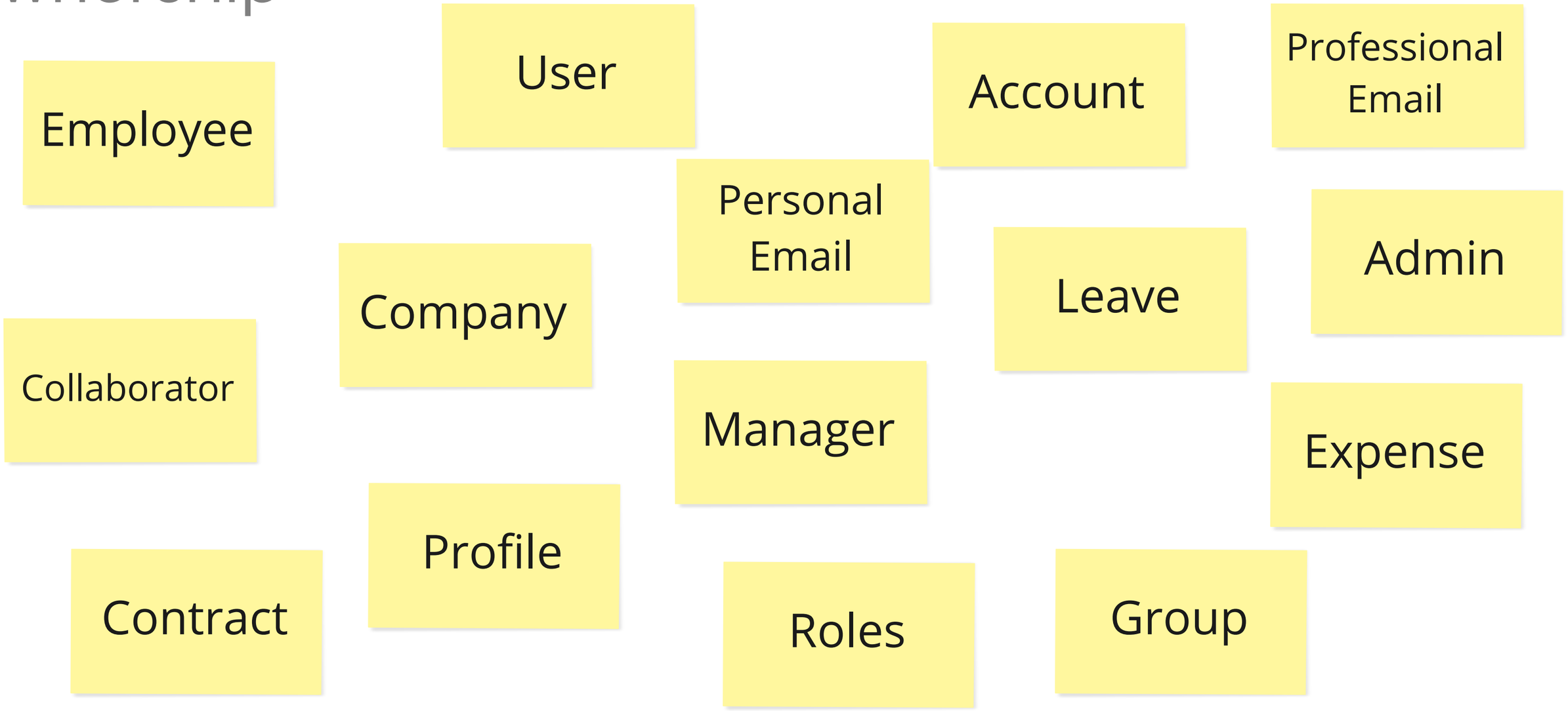
Context Cont'd

A lot of work has been done to define domains, and assign ownership to teams.

At the big picture-level things are clear.

But as teams are implementing modernization, and they are deeper into the details, more nuanced questions arise.

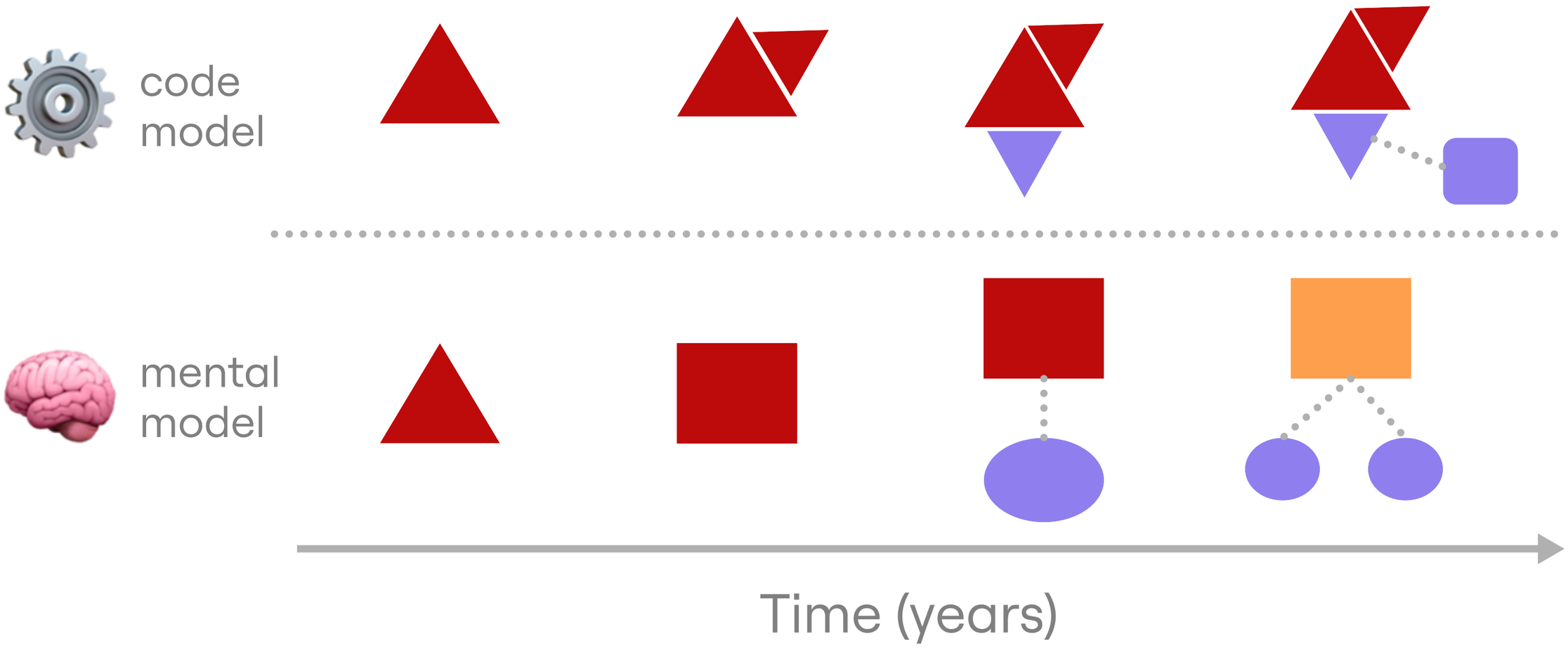
Seeking definitions, relationships, boundaries, ownership



Legacy-crystallized semantic drift

The concepts in our legacy become more and more misaligned with our mental model over time because legacy is hard to change

Modernization is a chance for semantic convergence

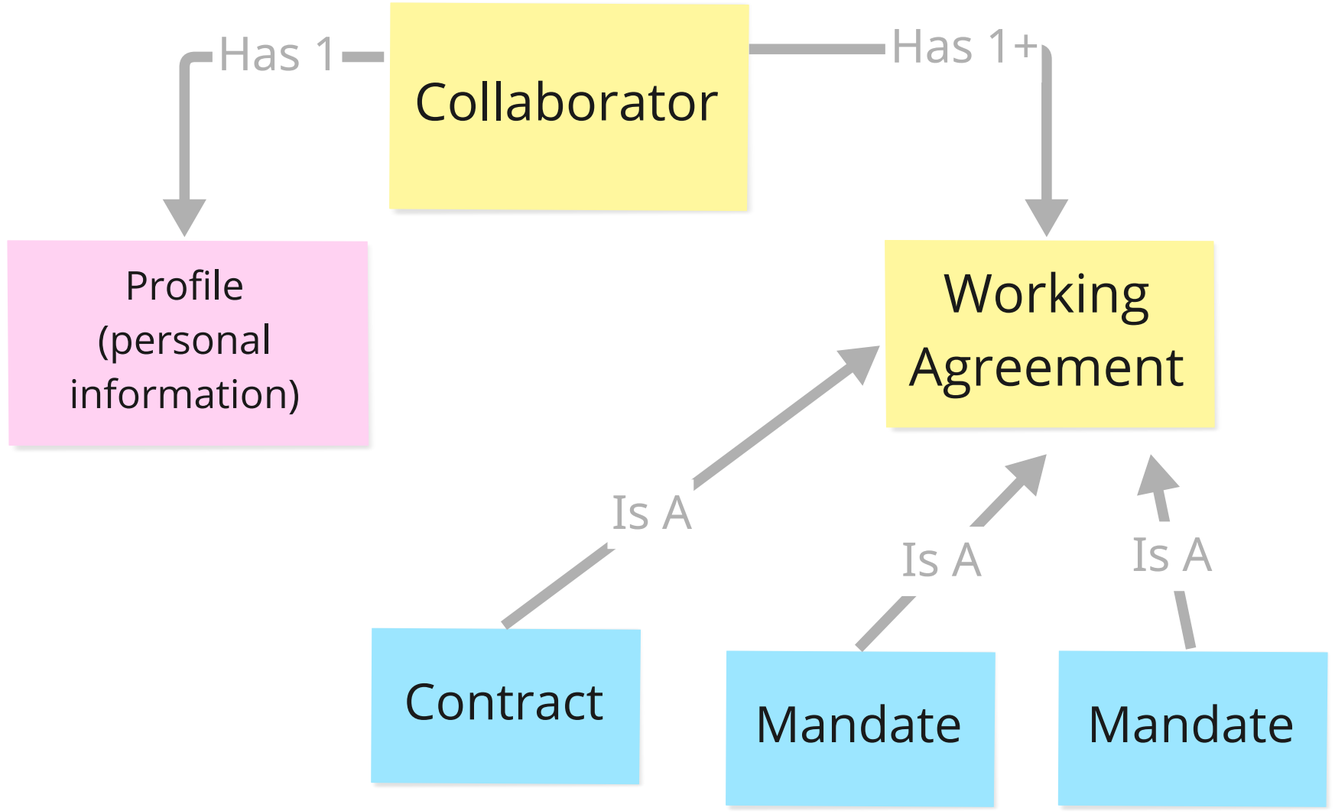


Drifting domain model – example

Model implemented in system

~~Employee~~ Contract
- personal information
- contract details
- contract type

1 domain expert's mental model



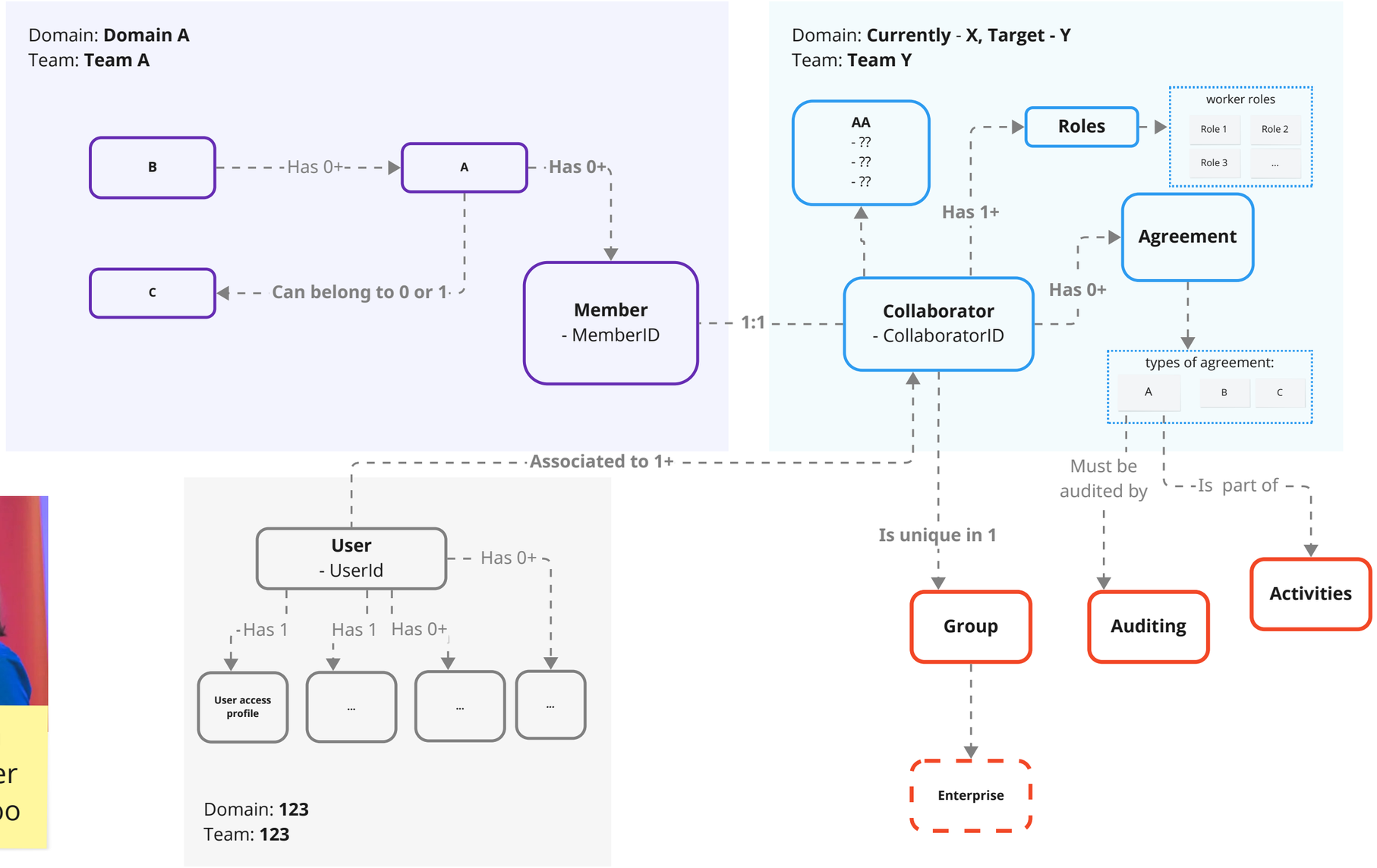
Challenge

As the model has shifted and different people and teams have developed their own mental model, defining a target model is not simple.

There are many contrasting opinions and beliefs about what things mean, how they should be modelled, and who should own them.

Building a domains map with entity relationships

Krisztina is a main character in this story too



How we built and agreed on the map

- Individual teams built proposals for their domains (with strong input from product & biz experts)
- Cross-team workshops (using DDD open hours)
- RFCs followed by ADRs for events
- Staff engineer community facilitated the process to ensure decisions were made and progress continued
- Raise awareness of target model in key meetings



Warning 3:
The unrelenting forces of anti-modernization gravity

I know we said modernization was important, but one of our biggest clients is asking for this new feature and we have to do it.

We can continue modernization in the 2nd half of the year after that.

-- VP Product

Apply the **anti-FAFO framework™** to combat the forces of anti-modernization gravity

Framing

Alignment on trade-offs

Fear & hope

Ongoing vigilance

Framing Modernization

Not like this

Product
vs
Tech

Like this !

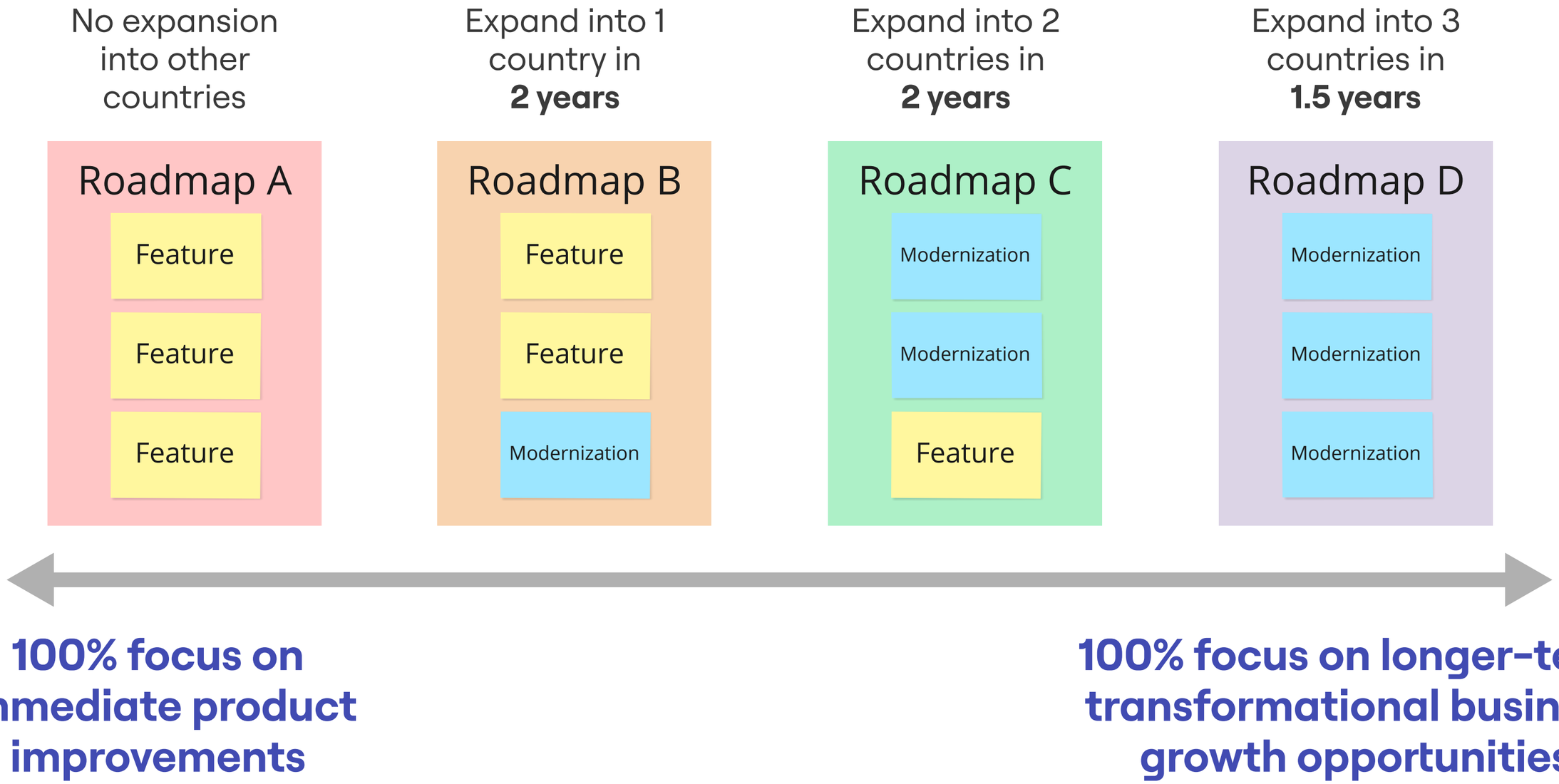
Immediate product improvements
vs
Longer-term transformational business growth capabilities

Invest in immediate product improvements

Significantly smaller but immediate ROI	Limited by what is possible in current system	Higher cost of change & maintenance	Slower innovation than competitors
Significantly greater ROI but with short-term compromise	Remove limitations / costs of current system	Lower costs to maintain and change	Faster innovation than competitors

Invest in longer-term transformational business growth capabilities

Align on trade-offs



Align on trade-offs

Are people hearing a consistent message from all members of leadership regarding priorities?

Are the CEO, CTO, CPO and others providing consistent messaging? Do they speak together?

Fear and hope

Constantly remind people how their choices will affect the future of the company

I'm really sorry that building this simple feature is not possible right now. But the modernization work we are doing will make this type of improvement possible within 1 year

Hope: new things will soon be possible if we stay committed

These production incidents and customer bugs will never go away if we keep deprioritizing modernization.

Fear: consequences of not modernizing

Ongoing vigilance

The forces of anti-modernization gravity are always present - every time you make a decision about what to work on

We have an urgent request from an important client. Please just add this 1 feature to the current sprint!

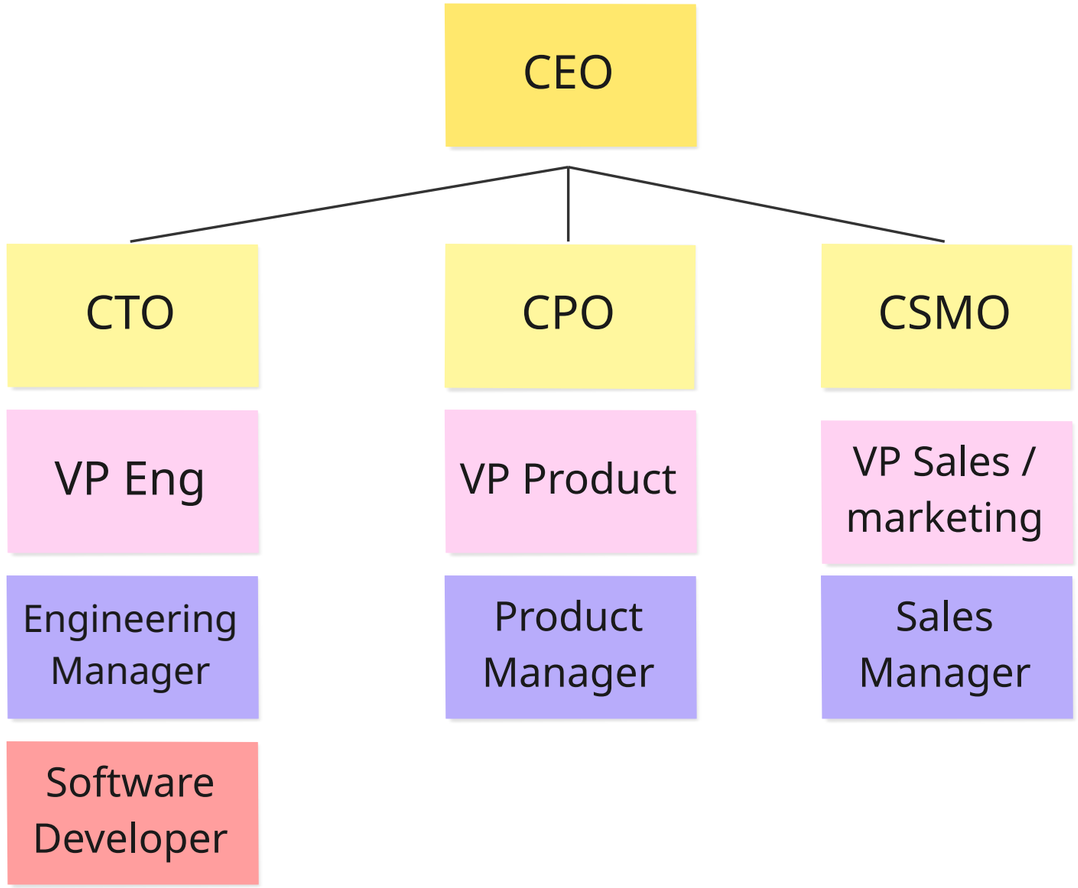
Ongoing vigilance

1. Every single prioritization decision counts - they all add up over time. Therefore, power point decks will not solve this problem.

2. Teams need to be educated, motivated and empowered to challenge prioritization decisions

Ongoing vigilance

Identify the source of misalignment and tension - it can come from anywhere



Warning 4:
Cognitive load

Modernization puts a lot of demands on your developers (and everyone else)

Learn new tech stack

Learn new infra

Learn new techniques and skills

Understand & change the legacy

Migrate / replace internal and external dependencies

work with domain experts to shape a new model (semantic divergence)

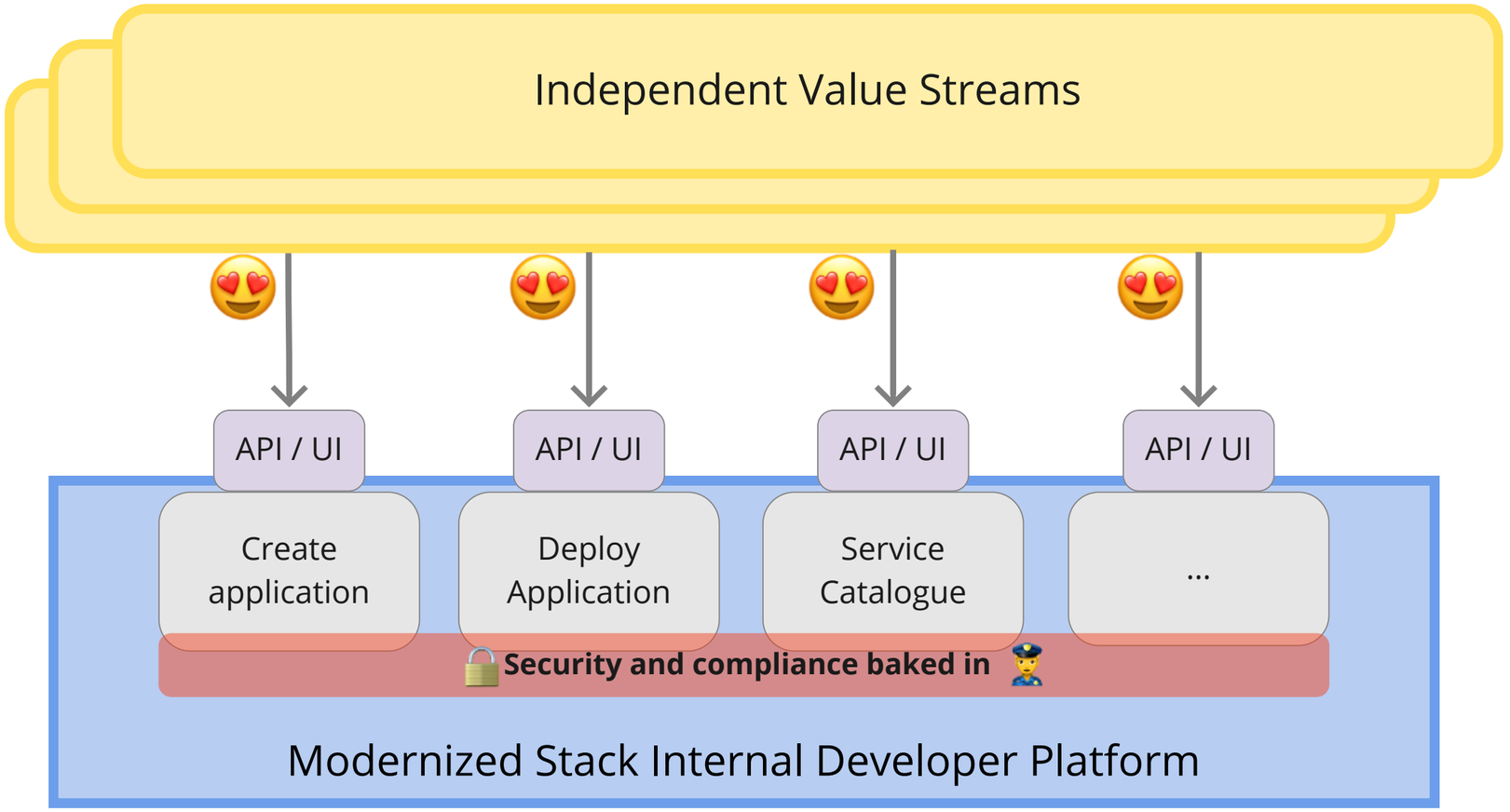
Incidents & bugs

Handle new and old running in paralle

Support other teams

Build new features

Leverage internal platforms to reduce cognitive load of teams modernizing



- It should take **minutes** to set up a new production-ready application
- Platforms need to be **opinionated** - you will never please everyone
- Good platforms can **accelerate** modernization tremendously

A focus on decomplexification can be pivotal

Nelson Da Costa

Let's get together a few times per week as a whole team for a mobbing session

Modernization can be stressful,
Try to be kind and treat others
how you would want to be
treated in their situation.

Recap: How to survive the long journey of modernization and reap the tremendous benefits?

Who are your Krisztinas?

Krisztina Hirth,
Staff Architect

Who in your org will establish and support regular learning activities for the skills you need to succeed with modernization?

☠️ Without these people, you will not escape legacy migration death valley.

Find or hire your Krisztinas as early as possible.

Who are your decision facilitators?

Who are the people in your company that will facilitate the process of making key decisions, especially those that that span multiple teams?

If you don't have people (and guidelines / processes) who can facilitate the decision making process and ensure decisions get made, you will be taking all your vacations in legacy migration death valley

Don't FAFO with anti-modernization gravity

If modernization = product vs tech legacy migration death valley is your new permanent home.

Control the narrative, provide hope, but fight when needed.

How will you reduce cognitive load of modernization?

Focus deeply and intensely on good internal platforms, and ruthlessly pursue decomplexification… and you might not end up in the modernization death valley

Thanks, enjoy the conference