



Doing a LeSS Flip in two months

9 October 2025

Document Classification:
public

We are GreenFlux, the smart charging pioneer in the eMobility industry.

13.6B

Electric km
powered

2.26B

kWh provided via
charging

1M

Roaming charging
points

106M

Charge sessions
completed

14y

Pioneer in Smart
Charging

Our single platform empowers the most ambitious
players to face the complexity and challenges of EV
charging head on.

Driving the Change



Ralph Hendriks
Director of engineering



Gianluca Cescon
Director of Product



Wouter Horsman
External Consultant



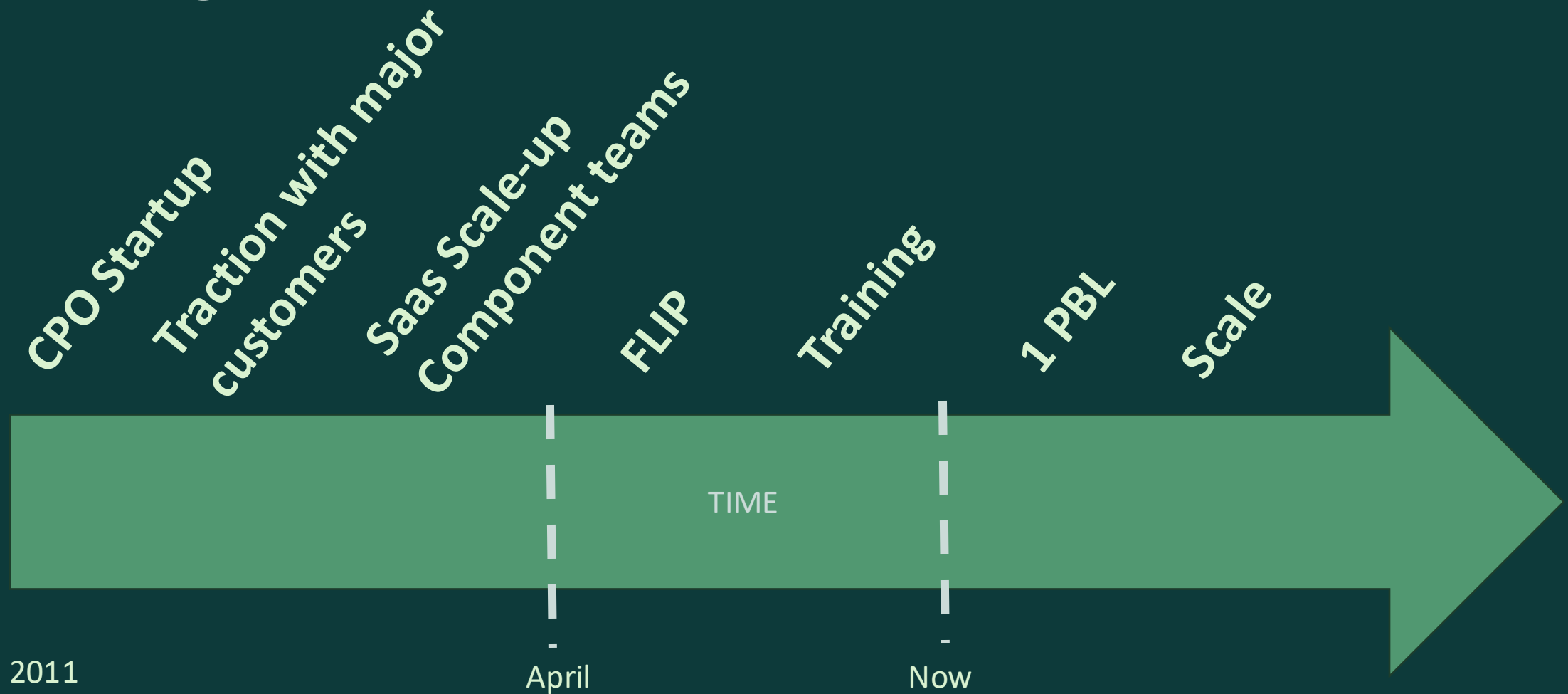
Roland Flemm
External Consultant

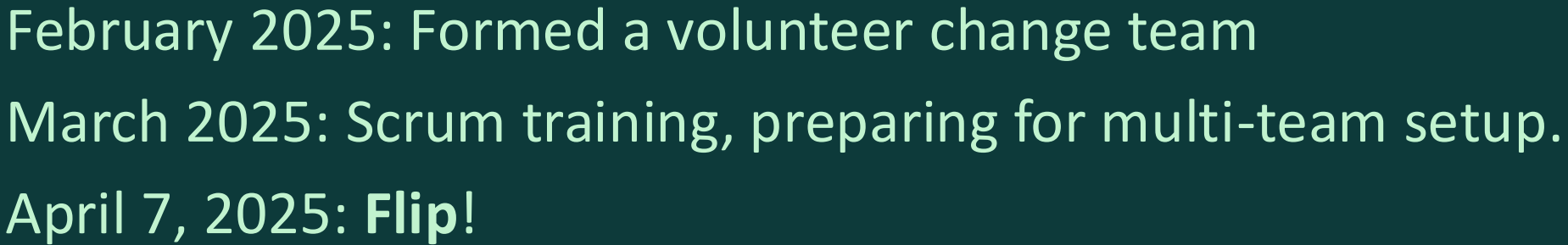
A Shout-out to Jurgen de Smet and Bas Vodde for their contribution.
Thanks for the Managing Directors Suthalan Ganes and Marc Erkens for their trust.

Results – 6 months Post-flip

- More adaptability.
- Teams think in customer problems.
- Reduced business risk (less SPOFs).
- Business continuity has improved.
- Transparency on our abilities.
- New hiring approach.
- A new product strategy with impact-driven roadmap.
- Adopting one PBL.

Why did we need a Change?





March 2025: Scrum training, preparing for multi-team setup.

April 7, 2025: Flip!

A two-month plan

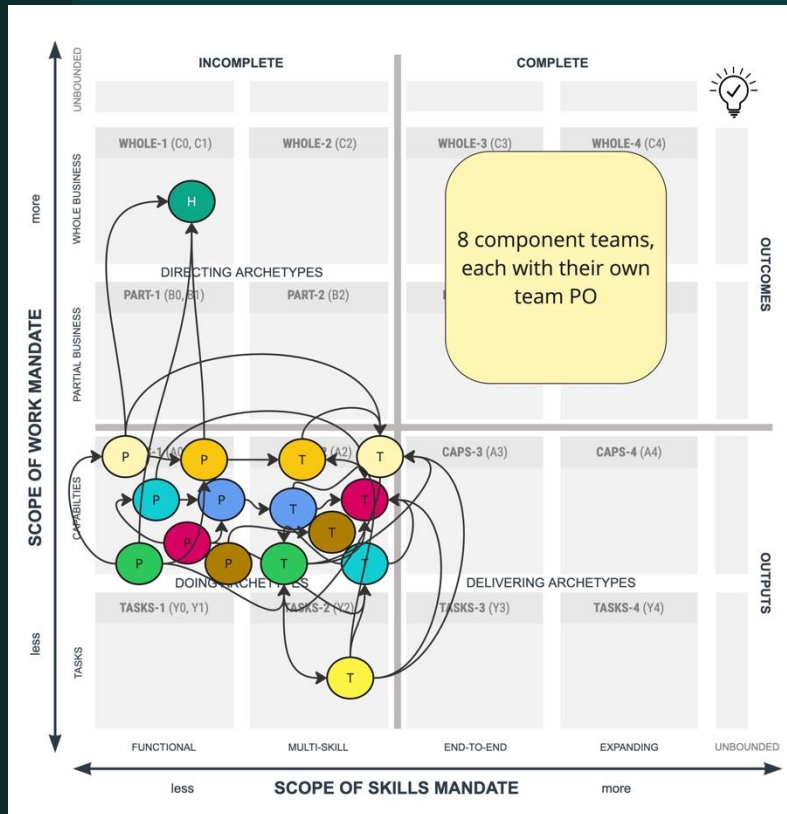
Takeaways

- A change backlog to know the minimum conditions for the Flip.
- A change backlog is not a checklist.
- Having no flip-experience is ok.
- Selling the WHY was easy.
- The HOW was hard.
- We never referred to LeSS.
- LeSS training 3months after the flip.



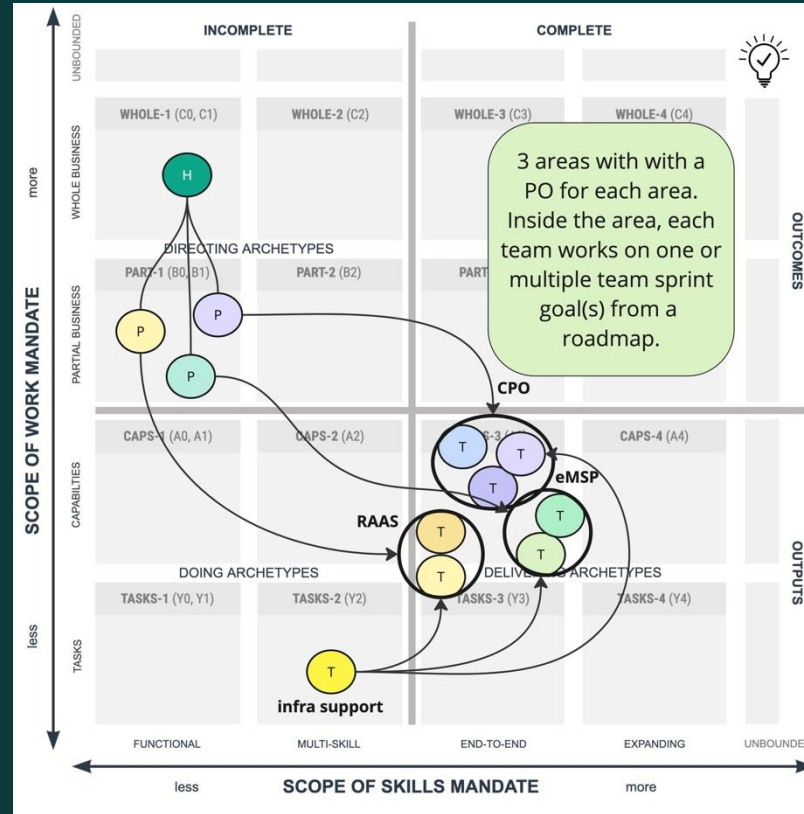
Design Choices

old design



Resource Topology

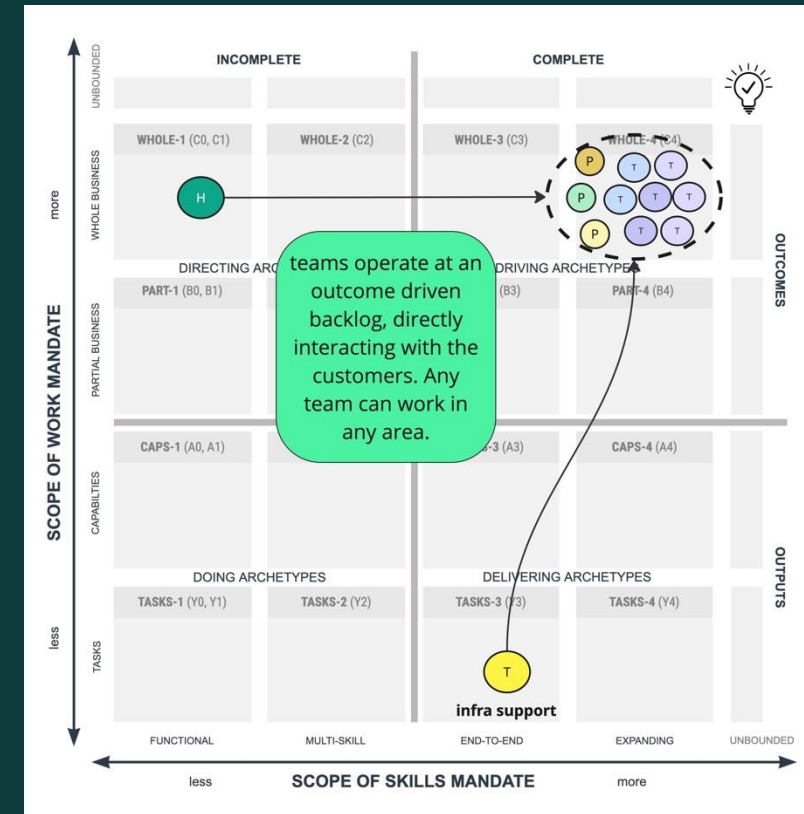
sept 2025



Delivery Topology
(Something eLSe)

ORG TOPOLOGIES

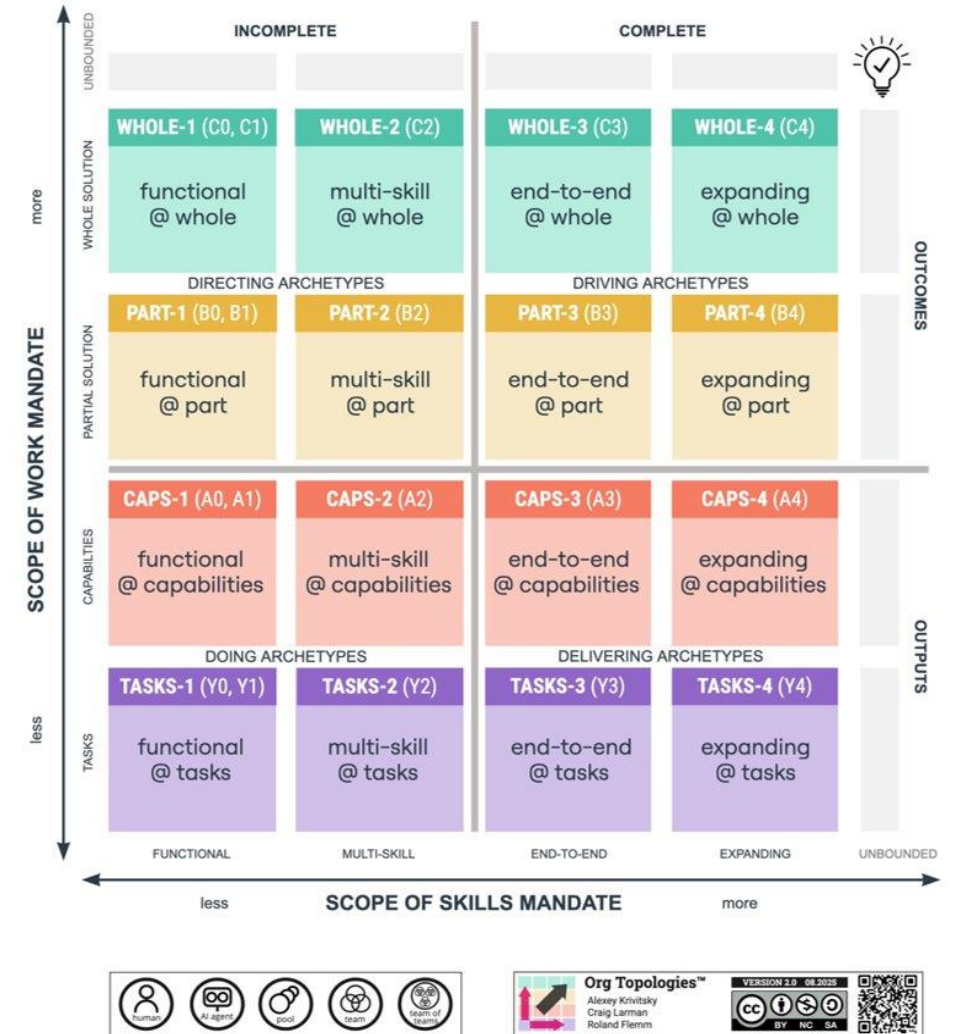
soon



Adaptive Topology
(LeSS)

Design Choices Takeaways

- Start with 3 product areas was a good idea for this organization:
 - Helped to get the change going.
 - Get commitment from senior leadership.
 - Supported maturing the PO-organization.
 - Made us learn the downsides of local optimization.
- Prefer Colocation
- Org Topologies helps



The Roadmap

- Roadmap was mapped on the old component structure:
 - contained fixed commitments.
 - was a list of projects.
 - reflected a Project mindset
- Moving towards Product Thinking:
 - Increased customer interaction through incremental releases.
 - Reduced the "deadline-focus" of the customer.

Roadmap Takeaways

- Executing the old roadmap hinders the transformation: Throw it away!
- With an unchanged roadmap, the POs remain order-takers.
- Starting with a new roadmap means moving toward improved product development:
 - Strategy re-orientation
 - Room for innovation
 - Product thinking

FROM:



TO:



Multi-Learning

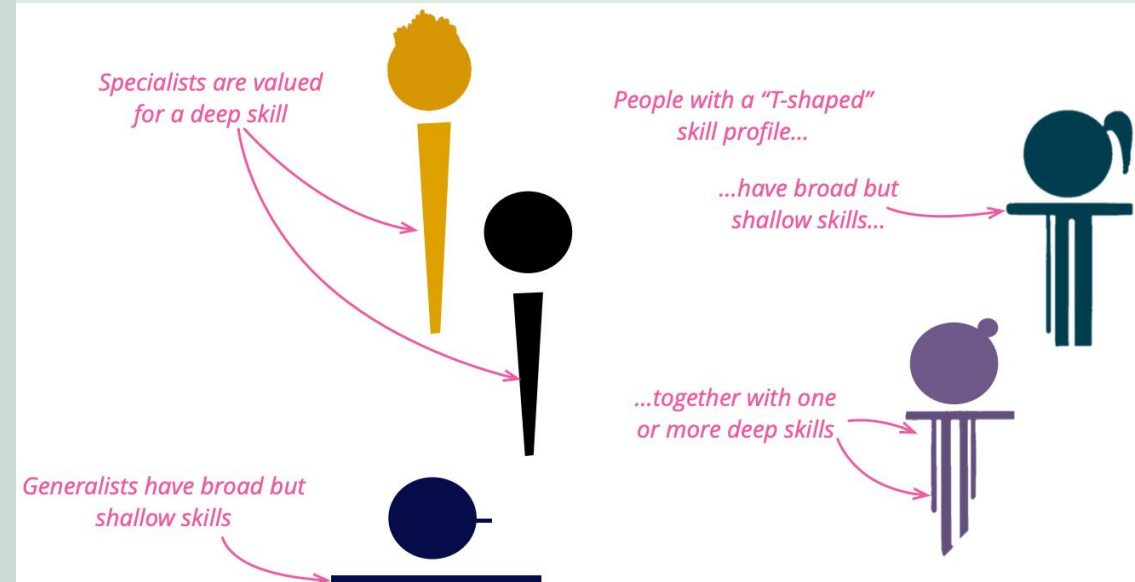
- Developer's belief: When I'm learning, I am not productive.
 - “Acquiring this knowledge takes me 6 sprints”
 - “I need a (formal) training”
 - “I need to know everything of everything”
- Some people feel underutilized and question the transformation.
- Knowledge transfer sessions turn into (recorded) lectures.
- Mobbing and Pairing gain traction.
- Disagreement on where to place UX in the org.



Multi Learning Takeaways

- People need to learn how to multi-learn.
- Knowledge-sharing through collaboration on real work.
- Renaming everybody to Product Developer can lead to resistance.
- Adding new M-shaped hires with a learning mindset works.
- Management needs to set clear expectations.
- Prepare and provide multi-learning pathways.
- Expertise is still a good thing.
- Align org design and reporting lines.

"Expert Generalists" (Martin Fowler)



Component Mentors

- Component mentors act as guardrails for quality.
- The architectural structure "does not flip".
- Component Mentors came under pressure:
 - Component mentors are most senior developers
 - Also need to support their colleagues for general concepts
 - Too many people need the same component mentors at the same time

Component Mentors Takeaways

- Component mentorship must be a temporary construct.
- Component mentors need to be coached actively:
 - Reducing complexity is the primary goal
 - Mentoring is not "being the architectural boss"
 - Prepare them for the time it takes to learn
 - Train them to be a mentor

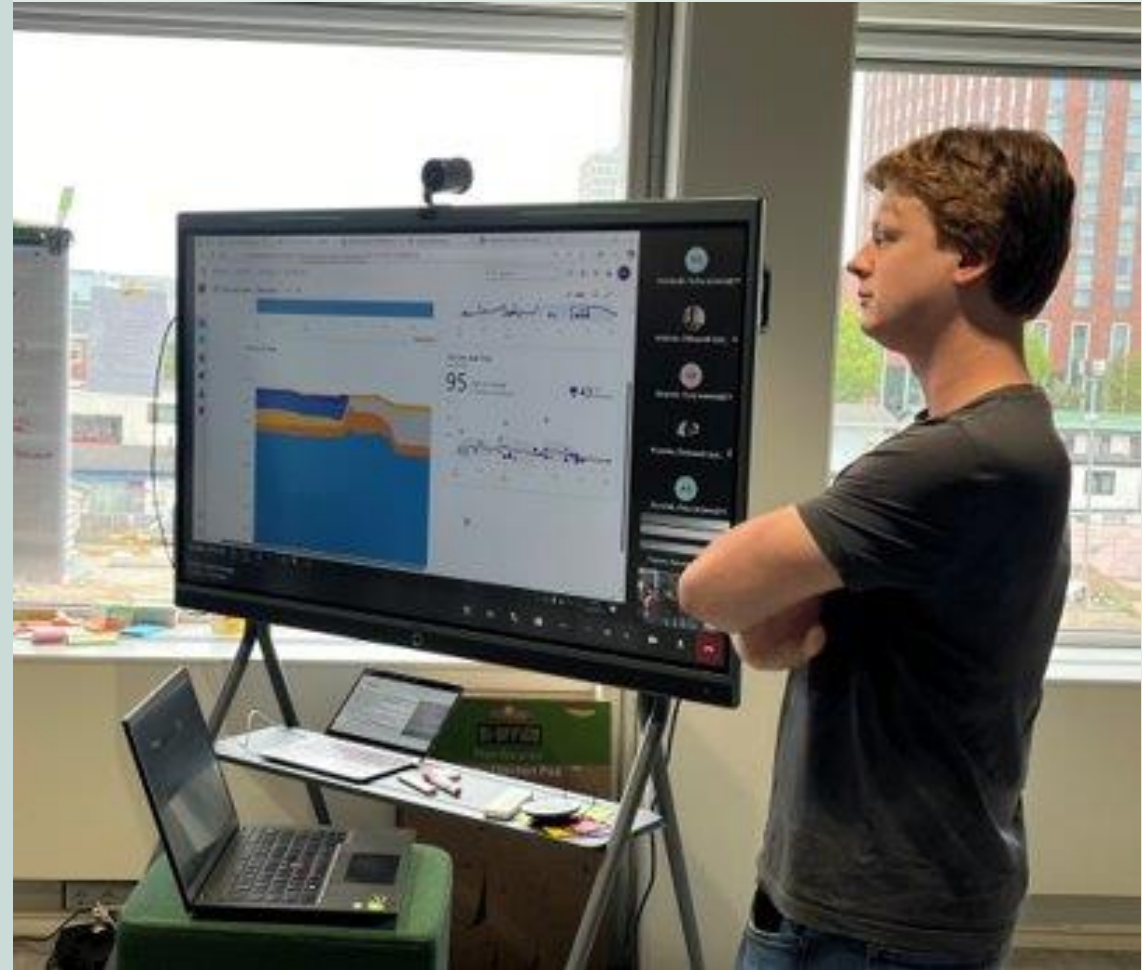


Technical Debt

- Component teams:
 - make tech debt less visible.
 - lead to different standards.
 - lead to duplicate solution implementations.
- Zero resistance to opening repositories.
- People thinking in big rewrites is also (tech) debt.
- Prioritizing tech debt on the PBL is not obvious.
- Ambitious DoD created *perceived* tech debt.

Technical Debt Takeaways

- Visualize the impact tech debt has on the product.
- Apply a "zero-bug policy" to actively reduce debt.
- Reduce tech debt OR new functionality is a false dichotomy.
- Duty team rotation for multi learning and quality awareness.
- Create a humble DoD that accurately represents the team's capabilities.



Results – 6 months Post-flip

- More adaptability.
- Teams think in customer problems.
- Reduced business risk (less SPOFs).
- Business continuity has improved.
- Transparency on our abilities.
- New hiring approach.
- A new product strategy with impact-driven roadmap.
- Adopting one PBL.

CONCLUSIONS

If we would do this again, we would...

we did this – we did not do this

- Choose simpler items to work on after the flip (initial PBL).
- Use the momentum to agree on the Flip date.
- Provide better support for multi-learning.
- First flip, then train (LeSS) later.
- Agree on formal reporting lines more clearly.
- Take Org Topologies training for mgt team gives a common language and supports the transformation.
- Agree on metrics (process and outcomes).

Back to the Pitch

- ✓ Is it a turbo-adoption?
- ✓ Is this an Adaptive Topology (LeSS) we are building?
- ✓ Or is the Delivery Topology (something eLSe) the best we can get?



Is this a Turbo-adoption?

No. You will never be fully ready to flip.

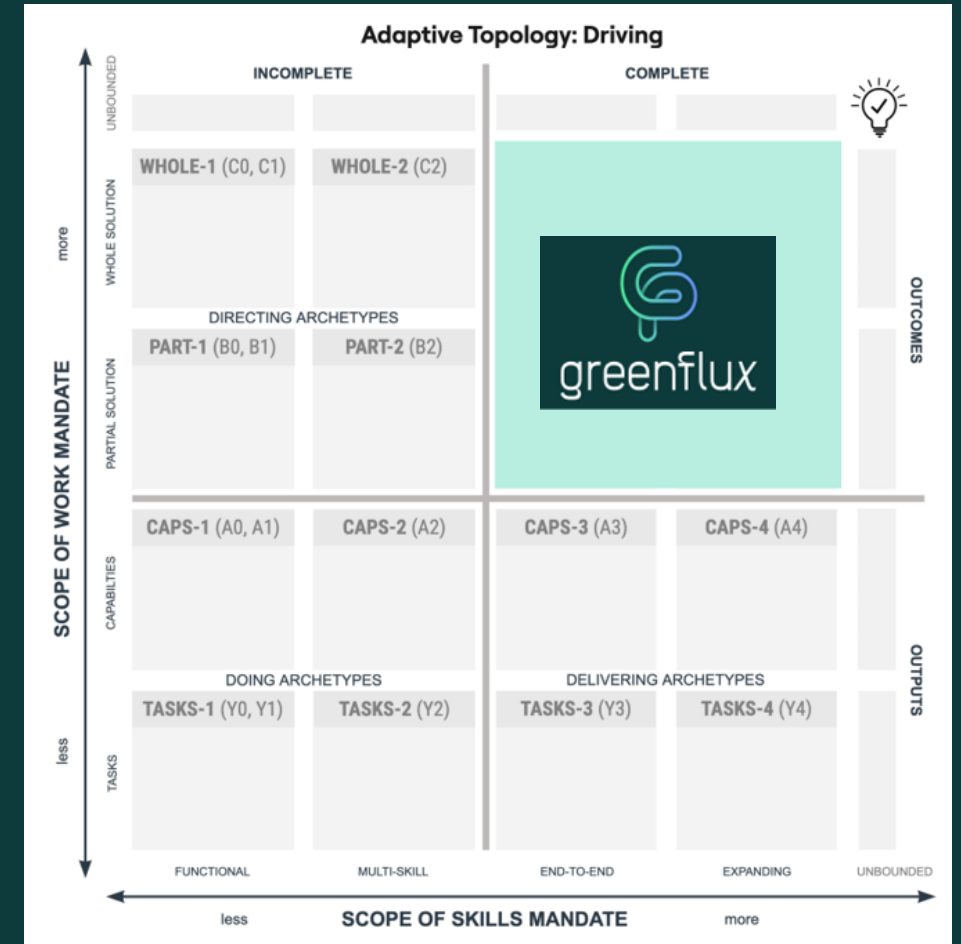


Is it LeSS (Adaptive Topology) or something eLSe?

Yes, it's LeSS, but not the first time right.

How did Org Topologies help?

1. Getting the change going (not selling LeSS).
2. Sharing the WHY of the change.
3. Creating a shared vocabulary.
4. Managing the evolution of the change.





Thank you!

