

Beyond the Spotify model

using Team Topologies for
organisation dynamics with software delivery

Matthew Skelton, Conflux

co-author of *Team Topologies* - @matthewpskelton

IPEXPO manchester
DIGITALTRANSFORMATIONEXPO

IPEXPO Mancs, 03 April 2019, Manchester



The Spotify Model



Limitations

Team Topologies



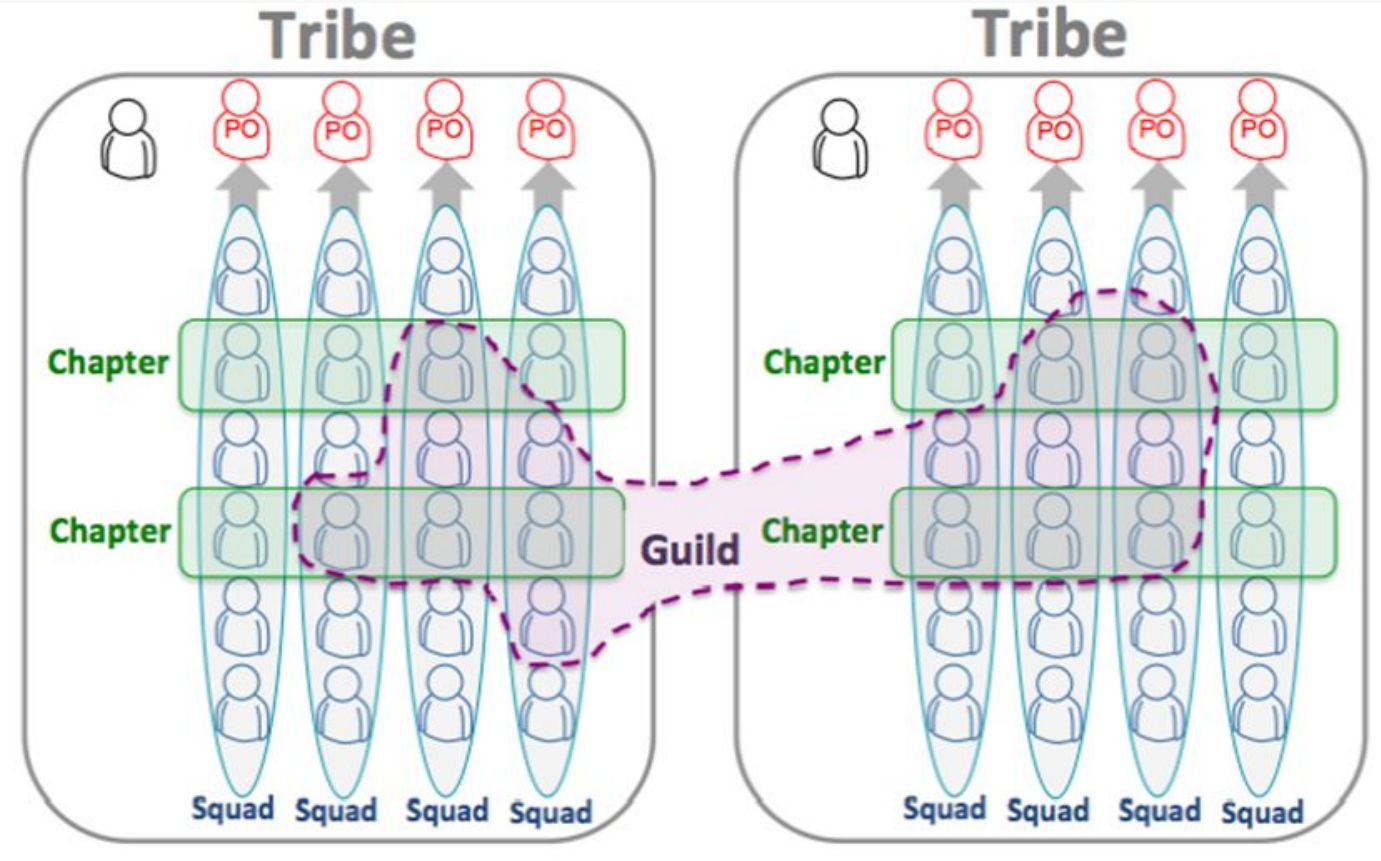
Getting started







The Spotify model of team design for software delivery



Henrik Kniberg & Anders Ivarsson, 2012

<https://blog.crisp.se/wp-content/uploads/2012/11/SpotifyScaling.pdf>

The Spotify model

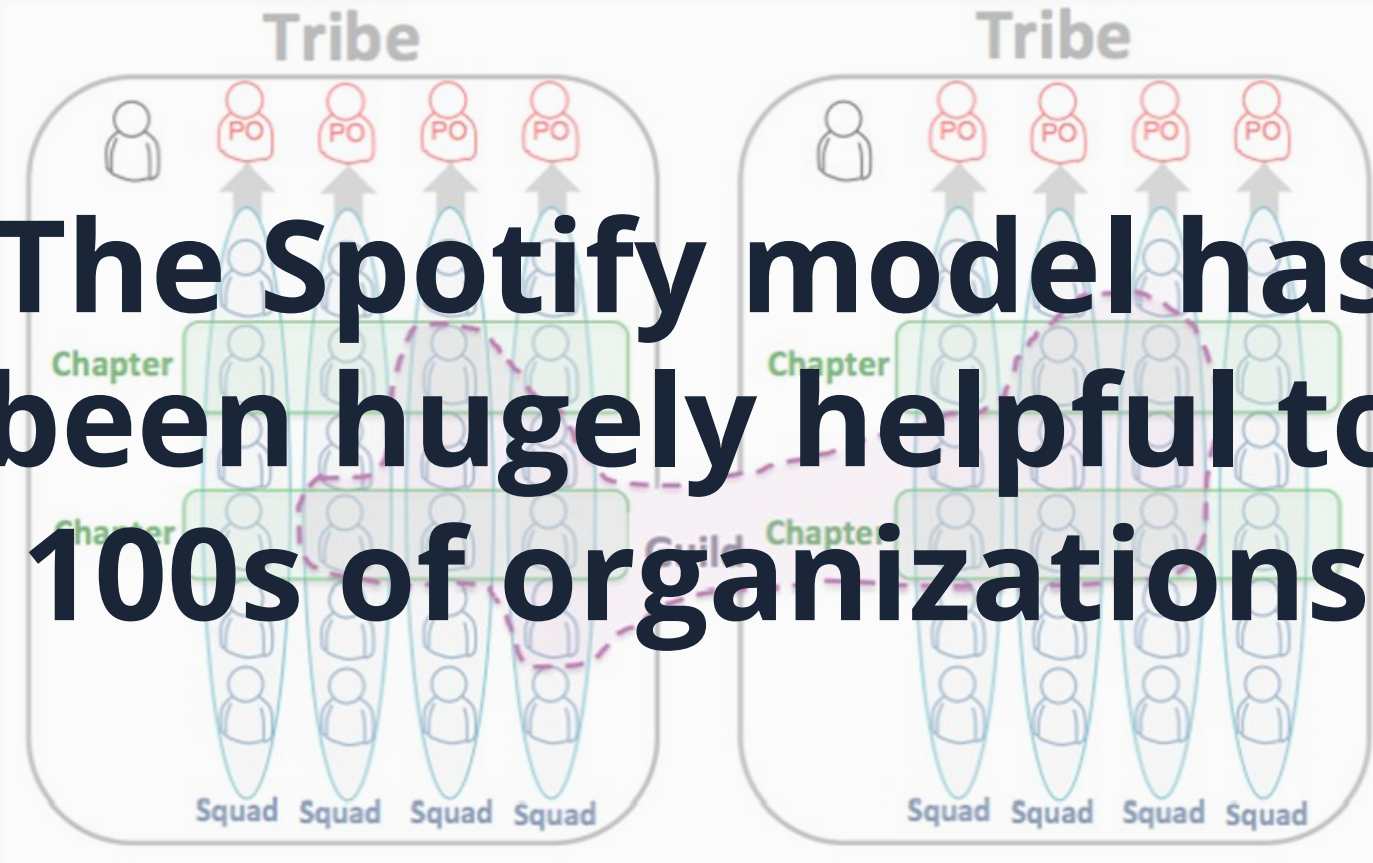
Squad: semi-autonomous delivery team

Tribe: family of Squads - related work

Chapter: line management within a Tribe

Guild: cross-Tribe interest/specialist group

**The Spotify model has
been hugely helpful to
100s of organizations**



The Spotify model helps to...

A photograph of several large, rusted metal pipes lying horizontally on a grassy field. The pipes are dark brown with visible rust and are arranged in a slightly overlapping manner. The background is a green, grassy hillside. The text "Encourage flow of change" is overlaid in white, bold, sans-serif font on the left side of the image.


**Encourage flow
of change**

The background image shows a construction or roadwork site. In the foreground, there are several stacks of orange traffic cones and yellow traffic cones with blue and red circular symbols. The ground is paved and covered with some dry leaves. In the background, there are more construction materials, a white van, and a person in a high-visibility vest. The text "Establish and clarify team responsibilities" is overlaid in white, bold, sans-serif font.

**Establish and
clarify team
responsibilities**



**Promote good
kinds of team
collaboration**

A photograph of a rustic wooden plank bridge crossing a small stream. The bridge is made of several weathered wooden planks. The stream is filled with water and large, smooth, light-colored rocks. The background is a dense forest with green foliage. The text "Plan and budget for cross-team enablers" is overlaid in white, bold, sans-serif font, centered on the image.

**Plan and budget
for cross-team
enablers**

The Spotify model helps to

Encourage flow of change

Establish and clarify team responsibilities

Promote good kinds of team collaboration

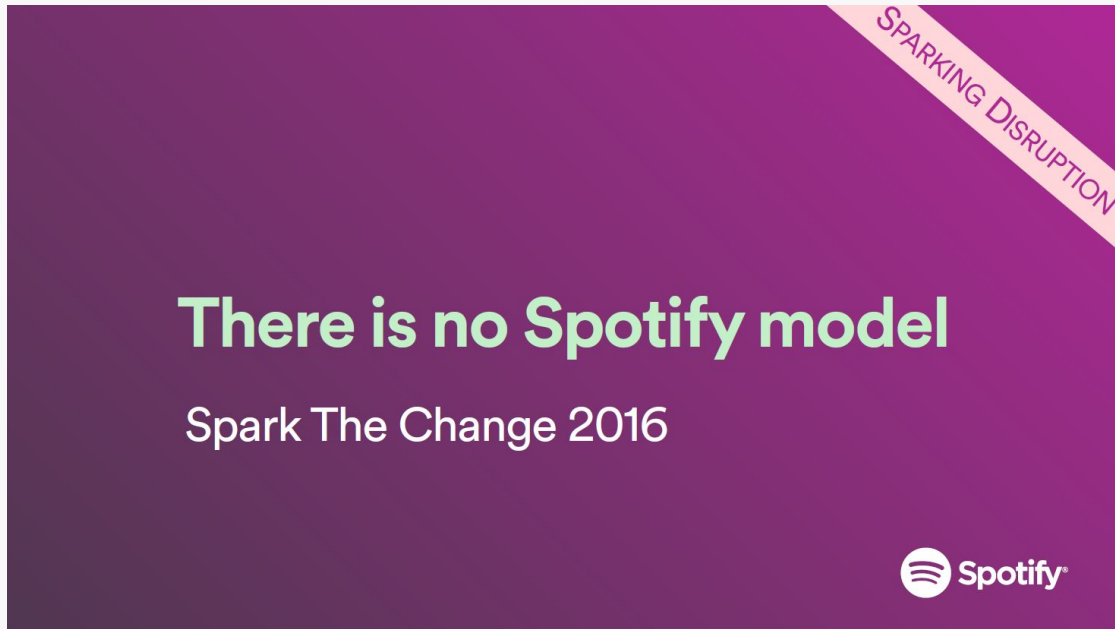
Plan and budget for cross-team enablers

Limitations of the Spotify model

“This article is only a snapshot of our current way of working - a journey in progress, not a journey completed. By the time you read this, things have already changed.”

- Kniberg & Ivarsson

There is No Spotify Model

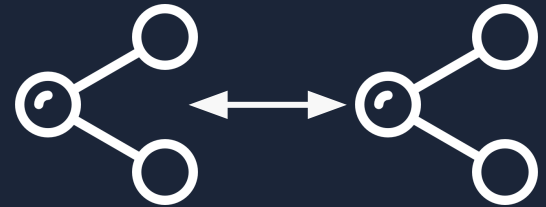


Marcin Floryan, 2016
<https://www.infoq.com/presentations/spotify-culture-stc>

Software sizing and cognitive load



Heuristics for Conway's Law



Patterns for team interactions



Triggers for change and evolution



We also need to address

Software sizing and cognitive load

Heuristics for Conway's Law

Patterns for team interactions

Triggers for change and evolution

Team Topologies

topology

the way in which constituent parts are
interrelated or arranged

Greek: *τοπολογία* (τόπος == 'place')

Team Topologies

Research over 5 years across multiple industry sectors

Informed by 50+ peer-reviewed journal articles

30+ client organizations - consulting and training since 2013 with orgs in CN, EU, IN, US, UK, +

Book: 12+ case studies from well-known organizations

Origins - DevOps Topologies

The screenshot shows the 'DevOps Topologies' website. The top navigation bar includes 'Anti-Types' and 'Team Topologies'. The main content area is divided into two sections. The first section, 'Anti-Type B: DevOps Team Silo', features a Venn diagram with three overlapping circles: 'Dev' (orange), 'DevOps' (pink), and 'Ops' (blue). The text explains that this topology typically results from a manager or exec deciding that teams 'need a bit of this DevOps thing' and starting a 'DevOps team' (probably full of people known as 'a DevOps'). The members of the DevOps team quickly form another silo, keeping Dev and Ops further apart than ever as they defend their corner, skills, and toolset from the 'clueless Devs' and 'dinosaur Ops' people. The second section, 'Anti-Type C: Dev Don't Need Ops', features a Venn diagram with two overlapping circles: 'Dev' (orange) and 'Ops' (blue). The text explains that this topology is borne of a combination of naivety and arrogance from developers and development managers, particularly when starting on new projects or systems. Assuming that Ops is now a thing of the past ('we have the Cloud now, right?'), the developers wildly underestimate the complexity and importance of operational skills and activities, and believe that they can do without them, or just cover them in spare hours. Below this, it states that such an Anti-Type C DevOps topology will probably end up needing either a Type 3 (Ops as Iaas) or a Type 4 (DevOps as a SaaS) topology when their software is deployed.

The screenshot shows the 'DevOps Topologies' website. The top navigation bar includes 'Anti-Types' and 'Team Topologies'. The main content area is divided into two sections. The first section, 'Type 2: Fully Shared Ops Responsibilities', features a Venn diagram with two overlapping circles: 'Dev' (orange) and 'Ops' (blue). The text explains that where operations people have been integrated in product development teams, we see a Type 2 topology. There is so little separation between Dev and Ops that all people are highly focused on a shared purpose; this is arguable a form of Type 1 (Dev and Ops Collaboration), but it has some special features. Organisations such as Netflix and Facebook with effectively a single web-based product have achieved this Type 2 topology, but I think it's probably not hugely applicable outside a narrow product focus, because the budgetary constraints and context-switching typically present in an organisation with multiple product streams will probably force Dev and Ops further apart (say, back to a Type 1 model). This topology might also be called 'NoOps', as there is no distinct or visible Operations team (although the Netflix NoOps might also be Type 3 (Ops as Iaas)). The second section, 'Type 3: Ops as Infrastructure-as-a-Service (Platform)', features a Venn diagram with two overlapping circles: 'Dev' (orange) and 'Ops' (blue). The text explains that for organisations with a fairly traditional IT Operations department which cannot or will not change rapidly (enough), and for organisations who run all their applications in the public cloud (Amazon EC2, Rackspace, Azure, etc.), it probably helps to treat Operations as a team who simply provides the elastic infrastructure on which applications are deployed and run; the internal Ops team is thus directly equivalent to Amazon EC2, or Infrastructure-as-a-Service.

[CC BY-SA devopstopologies.com](https://devopstopologies.com)

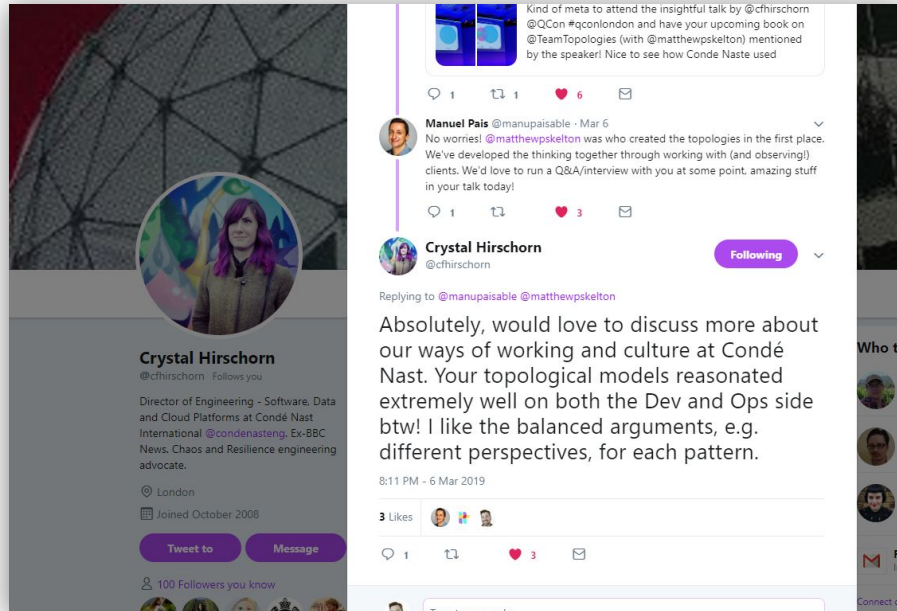




https://twitter.com/philip_pfo/status/999074792123740160

Philip Fisher-Ogden, Director of Engineering at Netflix:

“thanks for your insightful articulations of devops topologies. They inspired many discussions and helped us to think about what model Netflix teams could be/are using.”



<https://twitter.com/cfhirschorn/status/1103387659890819073>

Crystal Hirschorn, Director of Engineering at Condé Nast International

“Your topological models resonated extremely well on both the Dev and Ops side btw! I like the balanced arguments, e.g. different perspectives, for each pattern.”

Team Topologies

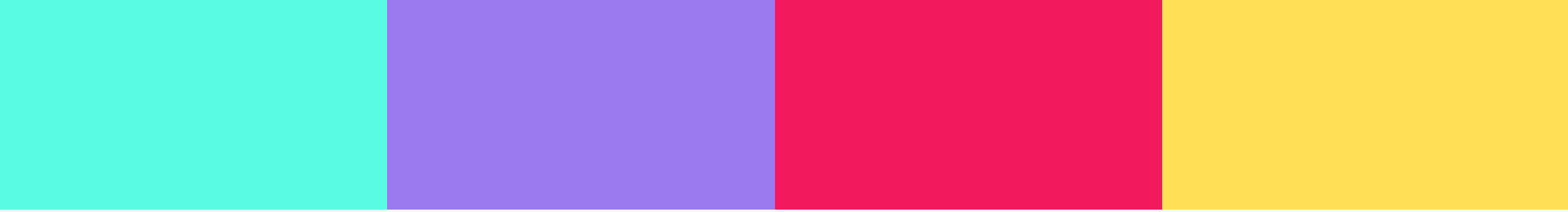
Organizing business and technology teams for fast flow

**Matthew Skelton and
Manuel Pais**

Publication date: Sept 2019
IT Revolution Press

Pre-order from Amazon.com:
<https://teamtopologies.com/book>





***“innovative tools and concepts for
structuring the next generation
digital operating model”***

Charles T. Betz, Principal Analyst, Forrester Research

Team Topologies for fast flow

**Team-first
Thinking**

Conway's Law

**Team
Interactions**

**Sensing for
Evolution**

Software sizing and cognitive load



Team-first Thinking



Team-first Thinking



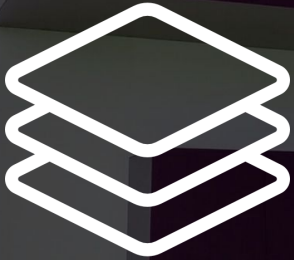
The team is the means of delivery

Team-first Thinking



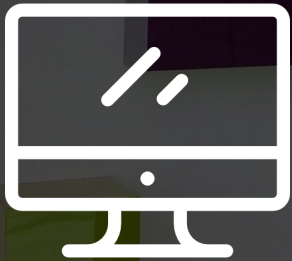
Design for team cognitive load

Team-first Thinking



Choose boundaries for team ownership

Team-first Thinking



Physical and digital workspace

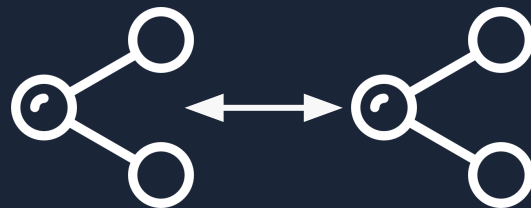
Team-first Thinking



The team is the means of delivery
Design for team cognitive load
Choose boundaries for team ownership
Physical and digital workspace

@matthewpskelton / @TeamTopologies

Heuristics for Conway's Law



Conway's Law



Conway's Law



Heuristic for 'natural' expected design



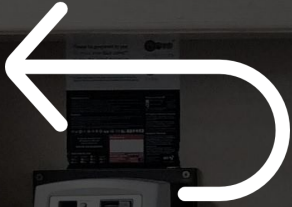
Conway's Law



Mirroring in tech system + human system

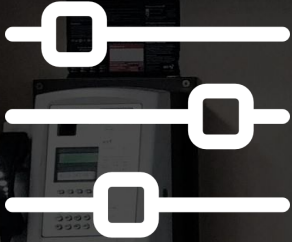


Conway's Law



Reverse Conway to mitigate worst effects

Conway's Law



Constraint on solution search space

Conway's Law



Heuristic for 'natural' expected design
Mirroring in tech system + human system
Reverse Conway to mitigate worst effects
Constraint on solution search space

Patterns for team interactions



Team Interactions



Team Interactions



3 defined Interaction Modes

Team Interactions



Collaboration: 2 teams working together



Team Interactions



X-as-a-Service: 1 provides, 1 consumes

Team Interactions



Facilitating: 1 team helps another

Team Interactions



3 defined Interaction Modes

Collaboration: 2 teams working together

X-as-a-Service: 1 provides, 1 consumes

Facilitating: 1 team helps another

4 fundamental topologies



Stream-aligned team



Enabling team

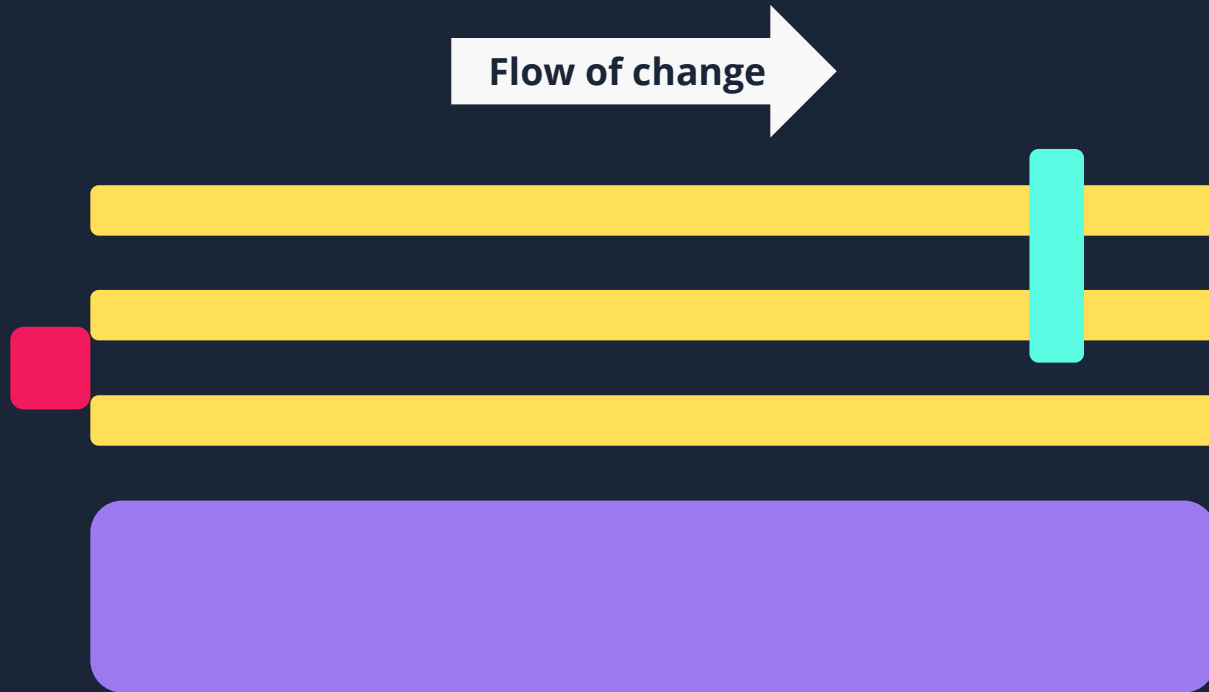


Complicated Subsystem team

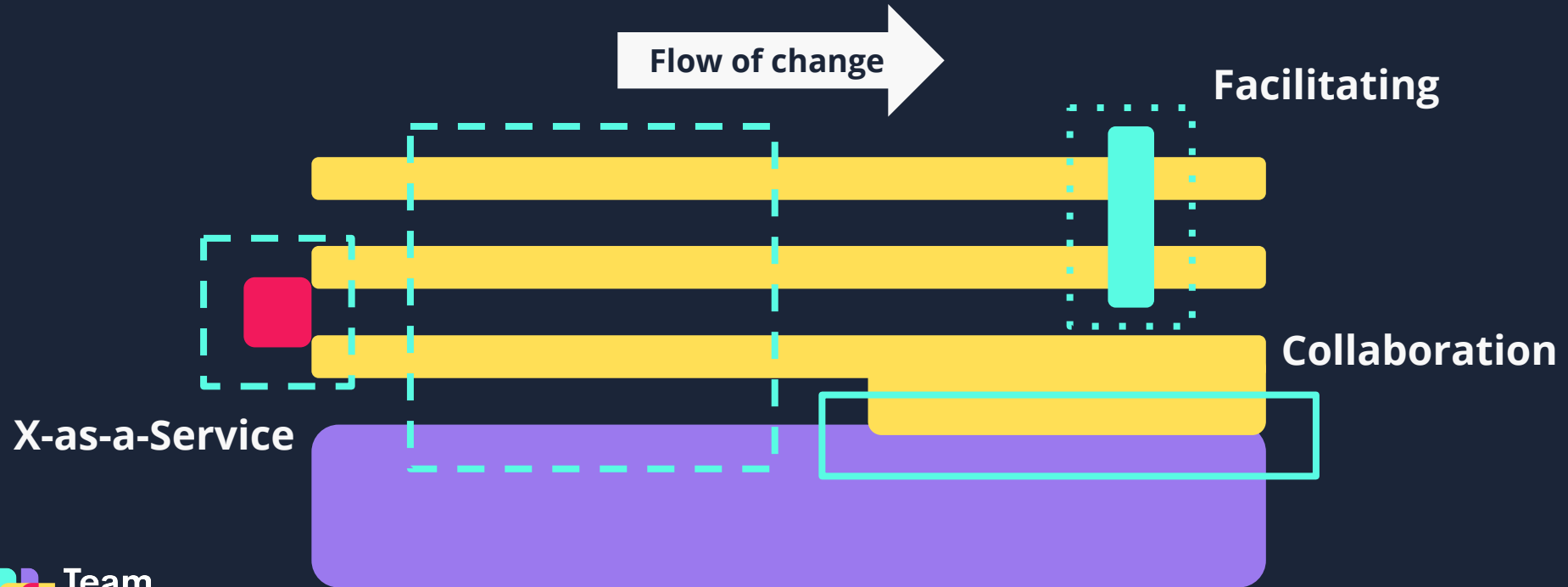


Platform team

4 fundamental topologies



3 core interaction modes



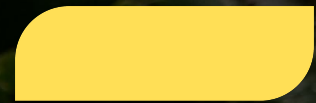
Triggers for change and evolution



Sensing for Evolution



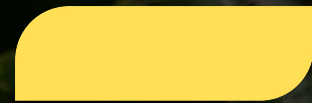
Sensing for Evolution



Not all teams in the org look the same



Sensing for Evolution



Discover, then push to Platform

Sensing for Evolution



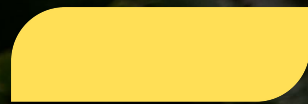
Awkward team interactions are signals

Sensing for Evolution



Evolve the org with changing ecosystem

Sensing for Evolution



Not all teams in the org look the same
Discover, then push to Platform
Awkward team interactions are signals
Evolve the org with changing ecosystem

Getting started with the Team Topologies approach

Getting started



Explicit cognitive load

Explicit cognitive load

How well can the team as a unit “grok” the systems they own and develop?

Push some things into a Platform?

Are skills or capabilities missing?

Getting started



Large Conway mismatches

Large Conway mismatches

Are there major mismatches between the team interactions and the required software / system architecture?

What could be easily adjusted?

Getting started



Team Interactions

Team Interactions

What would change if we adopted the 3 team interaction patterns?

Collaboration, X-as-a-Service, Facilitating

How would teams react & behave?

Getting started



Thinnest Viable Platform

Thinnest Viable Platform

How is your Platform defined?

What is the thinnest platform that could work?

What's needed to run and support it?

Getting started



Explicit cognitive load



Large Conway mismatches



Team Interactions



Thinnest Viable Platform

Review

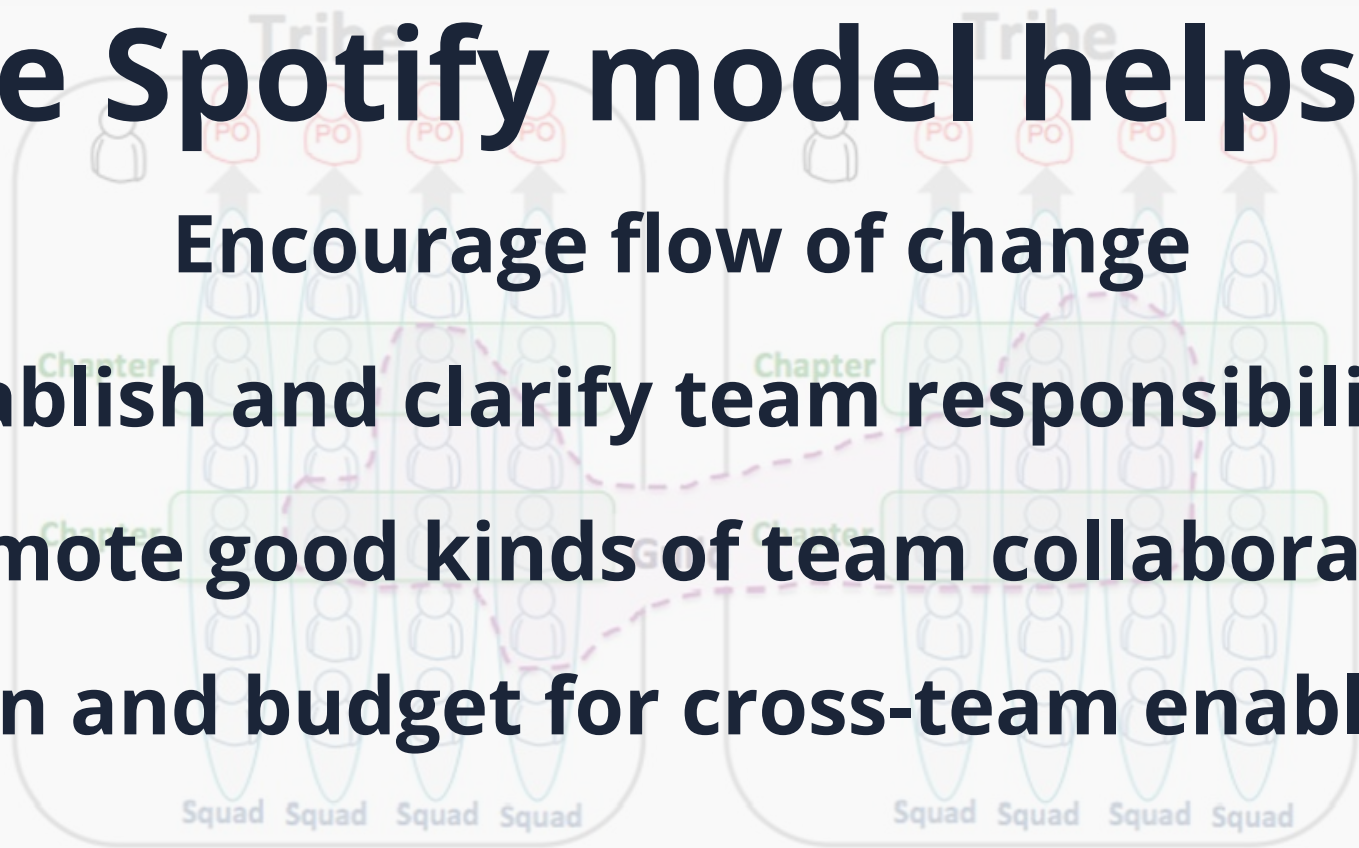
The Spotify model helps to

Encourage flow of change

Establish and clarify team responsibilities

Promote good kinds of team collaboration

Plan and budget for cross-team enablers



We also need to address



Software sizing and **cognitive load**



Heuristics for **Conway's Law**



Patterns for **team interactions**



Triggers for **change and evolution**

Team Topologies for fast flow

**Team-first
Thinking**

Conway's Law

**Team
Interactions**

**Sensing for
Evolution**

Getting started



Explicit cognitive load



Large Conway mismatches



Team Interactions



Thinnest Viable Platform

Sign up for news and tips:

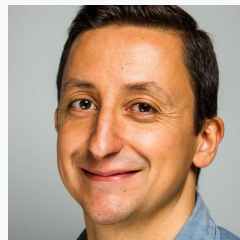
TeamTopologies.com

Thank you!

info@teamtopologies.com



Matthew Skelton, Conflux
@matthewpskelton



Manuel Pais, Independent
@manupaisable



Copyright © [Conflux Digital Ltd](#) 2018-2019. All rights reserved.
Registered in England and Wales, number 10890964

Icons made by [Freepick](#) from [www.flaticon.com](#) - used under license