

How and why to design your teams for modern software systems

Matthew Skelton | Skelton Thatcher Consulting
@matthewpskelton | skeltonthatcher.com



DevOps Enterprise Summit / @DOES_EUR / #DOES17
06 June 2017, London UK



Today

- Conway's Law (or heuristic)
- Cognitive Load for teams
- Real-world Team Topologies
- Guidelines for team design



About me

Matthew
Skelton

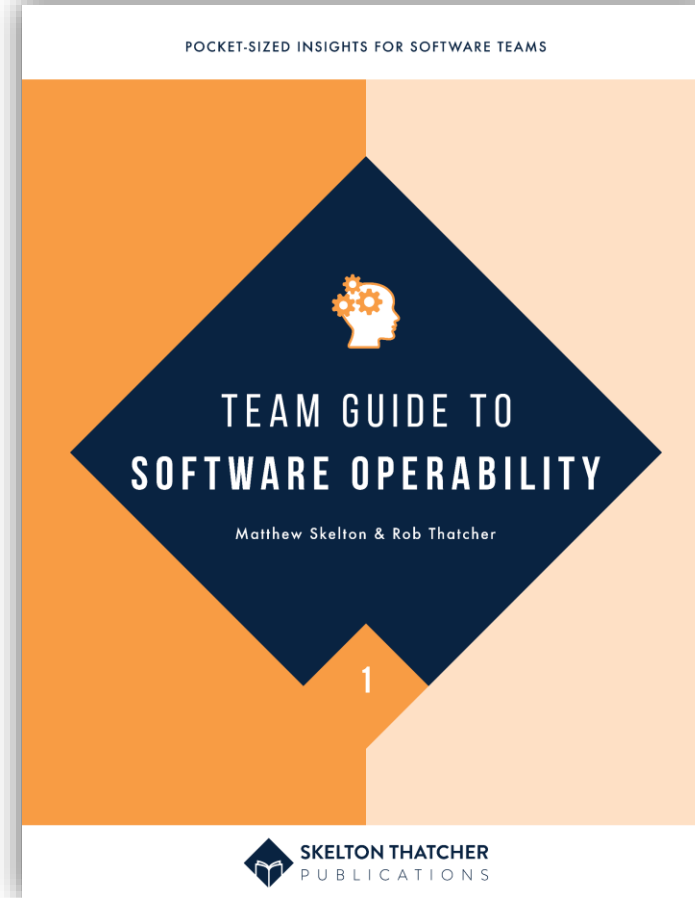
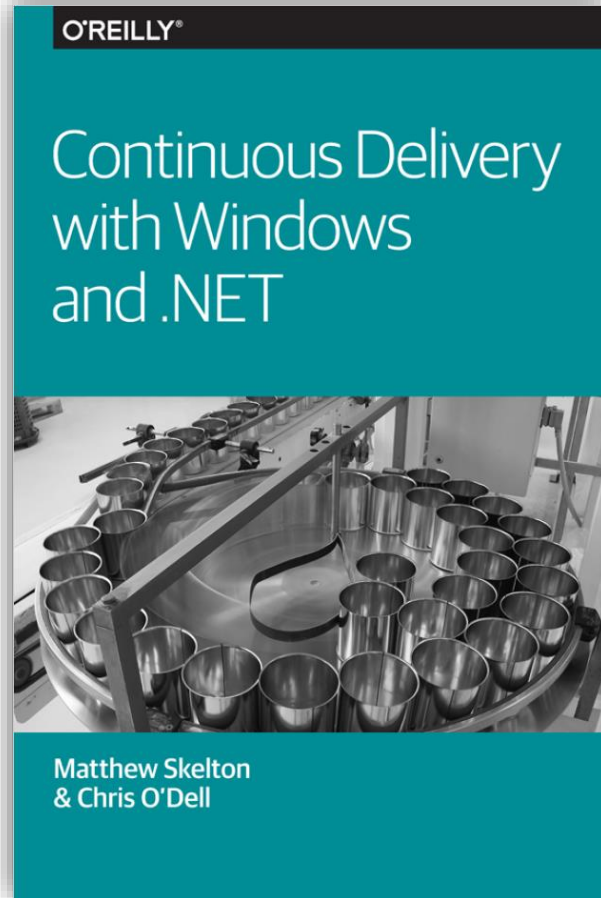
@matthewpskelton

Co-founder at
Skelton Thatcher Consulting
skeltonthatcher.com





Books





SKELTON THATCHER
CONSULTING

skeltonthatcher.com

Use code
"DOES17" for
25% discount

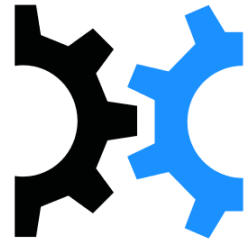


SKELTON THATCHER
EFFECTIVE SOFTWARE OPERATIONS

Organisation design for effective software systems - Tutorial / Workshop -
Sept 2017

📅 September 27th, 2017 📍 London, UK

<https://ti.to/skelton-thatcher-consulting/organisation-design-workshop-sept-2017>



SKELTON THATCHER

EFFECTIVE SOFTWARE OPERATIONS



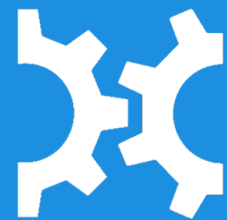
Team-first digital transformation

30+ organisations

UK, EU, DE, India, China

skeltonthatcher.com

We build modern capabilities
by mentoring your teams



SKELTON THATCHER
EFFECTIVE SOFTWARE OPERATIONS





How and why to design your teams for modern software systems



Safer, more rapid
changes to
software systems
(Business Agility)



TEAM



TEAM



TEAM

capabilities

appetite & aptitude

understanding

responsibilities



(assumption)

the team is stable, slowly changing,
and long-lived

#NoProjects



Conway's Law

(or Conway's Heuristic)



“organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations”

– Mel Conway, 1968



“if the architecture of the system and
the architecture of the organization
are at odds, the architecture of the
organization wins”

– Ruth Malan, 2008



“We find strong evidence to support the hypothesis that a product’s architecture tends to mirror the structure of the organization in which it is developed.”

– MacCormack et al, 2012



homomorphic force

(same)

(shape)

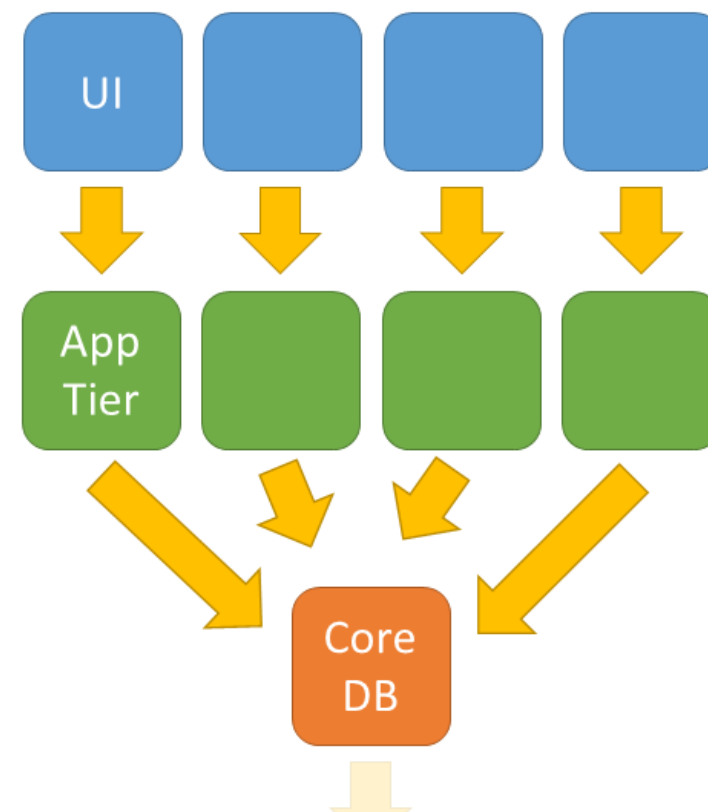
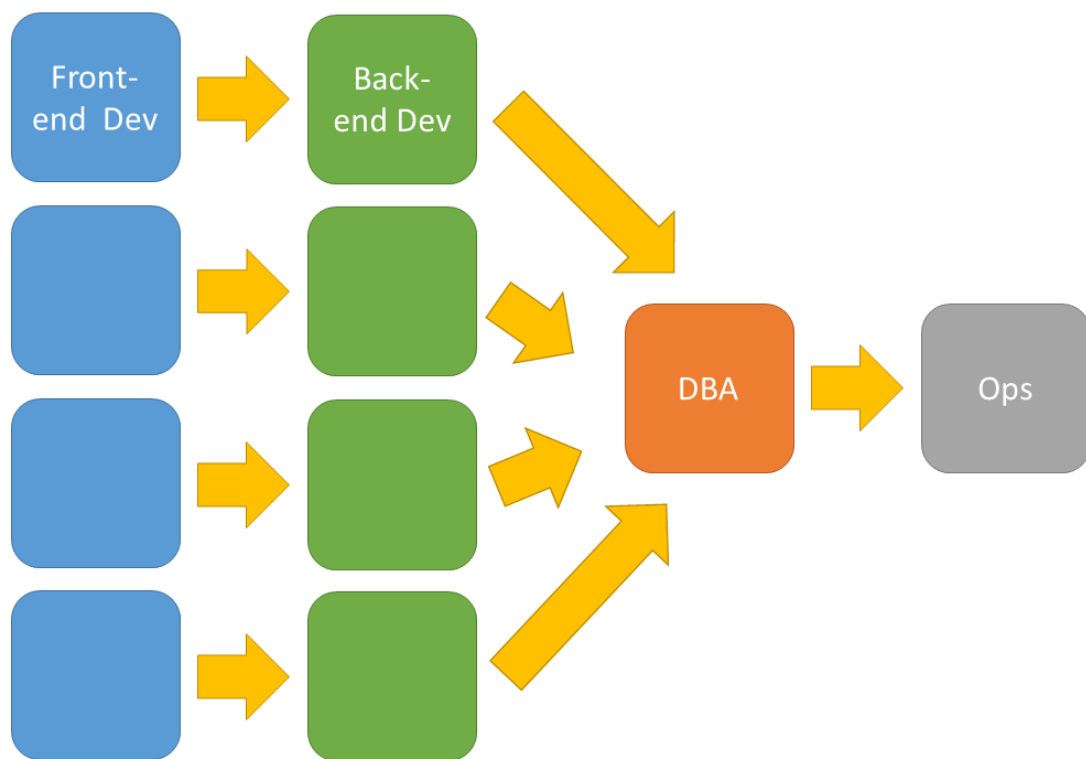
(#Conway \leftrightarrow #Yawnoc)

HT @allankellynet



**Front-end
developers**

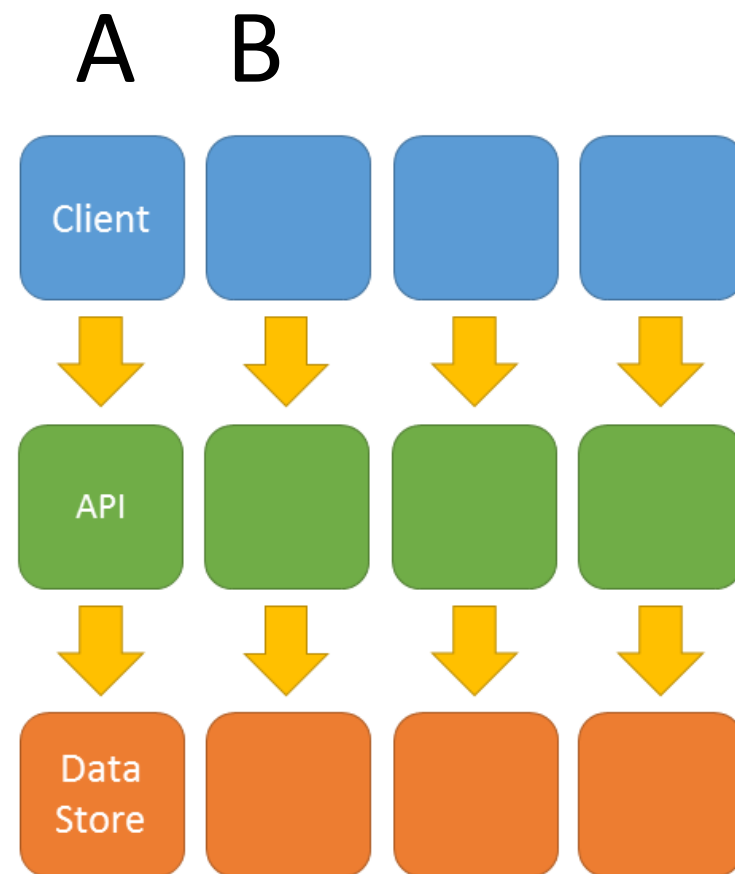
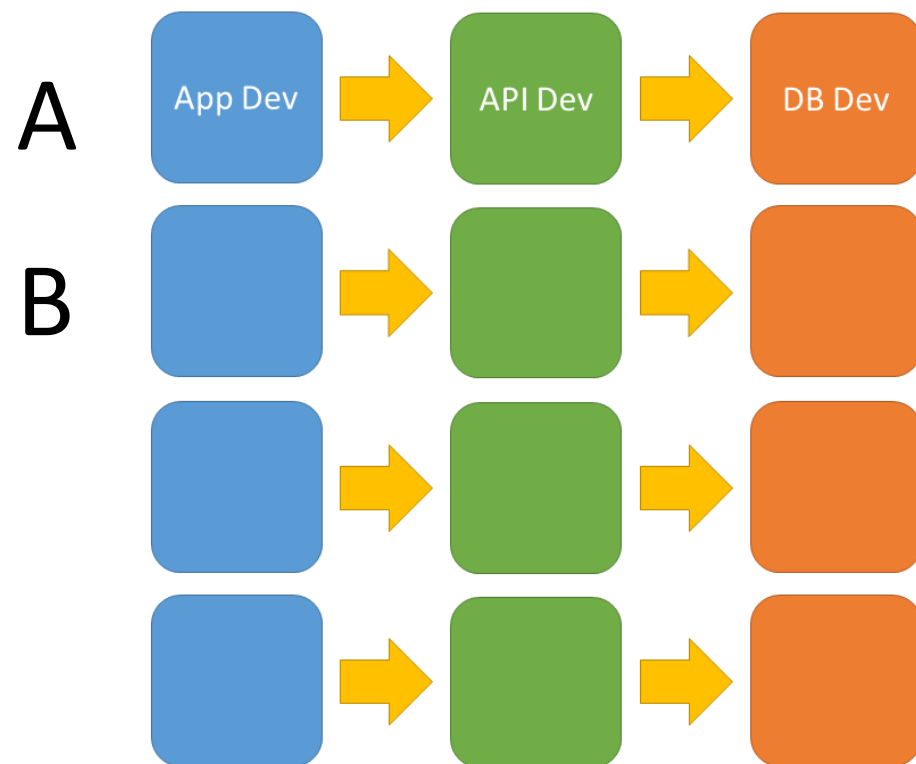
**Back-end
developers**





'Reverse Conway'

Tobbe Gyllebring (@drunkcod)







**Design the
organisation architecture
to produce the right
software architecture**



Cognitive Load for teams



Cognitive load

the total amount of
mental effort being used in
the working memory

(see Sweller, 1988)



Cognitive load

Intrinsic

Extraneous (Irrelevant)


Germane (Relevant)



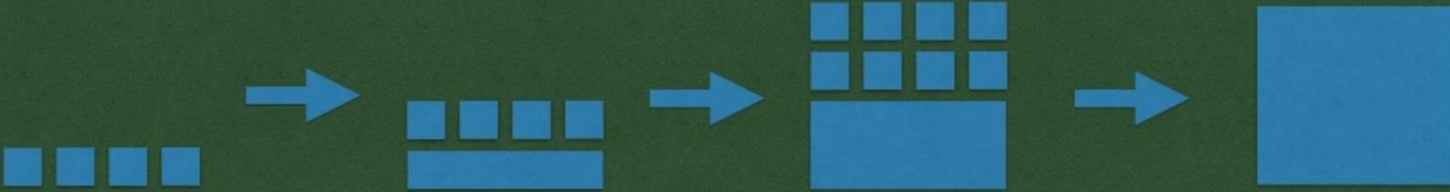
'Hacking Your Head': Jo Pearce

[@jdpearce](#)

Hacking Your Head : Managing Information Overload

 Intrinsic Load

Leverage the concept of chunking.



@jdpearce jopearce.co.uk

See <http://www.slideshare.net/JoPearce5/hacking-your-head-managing-information-overload-45-mix>



We have SCIENCE!



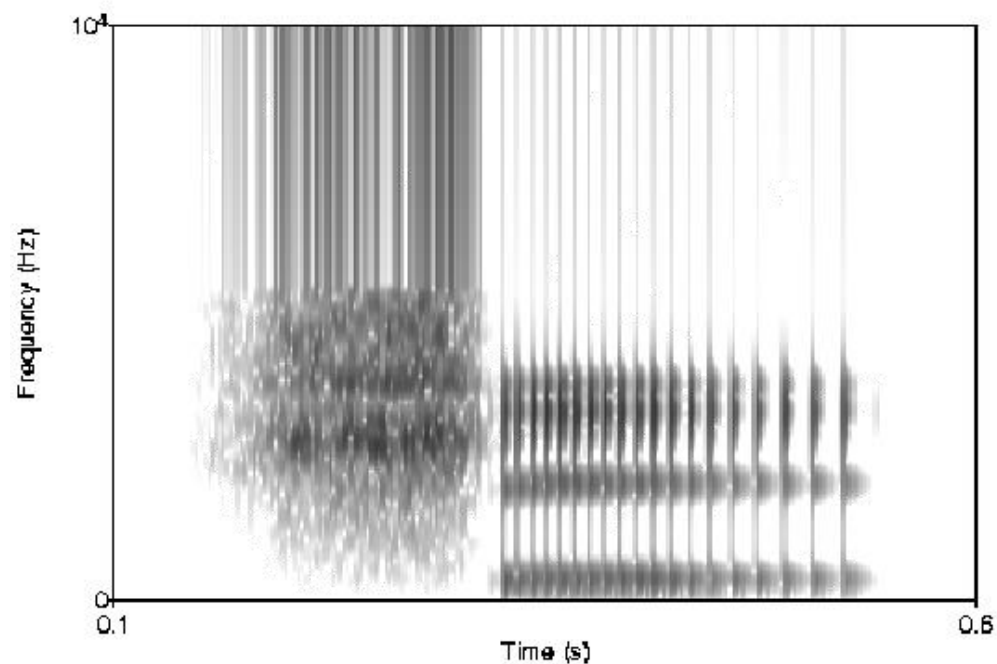
Science since 1988

- Driskell et al, 1999 'Does Stress Lead to a Loss of Team Perspective?' Group Dynamics: Theory, Research, and Practice 3, no. 4 (1999): 291.
- Fan et al, 2010 'Learning HMM-Based Cognitive Load Models for Supporting Human-Agent Teamwork'. Cognitive Systems Research 11, no. 1 (2010): 108–119.
- Ilgen & Hollenbeck, 1993 'Effective Team Performance under Stress and Normal Conditions: An Experimental Paradigm, Theory and Data for Studying Team Decision Making in Hierarchical Teams with Distributed Expertise'. DTIC Document, 1993.
- Johnston et al, 2002 'Application of Cognitive Load Theory to Developing a Measure of Team Decision Efficiency'. DTIC Document, 2002.
- Sweller, John, 1994 'Cognitive Load Theory, Learning Difficulty, and Instructional Design'. Learning and Instruction 4 (1994): 295–312.
- Sweller, John, 1988. 'Cognitive Load during Problem Solving: Effects on Learning'. Cognitive Science 12, no. 2 (1988): 257–285.

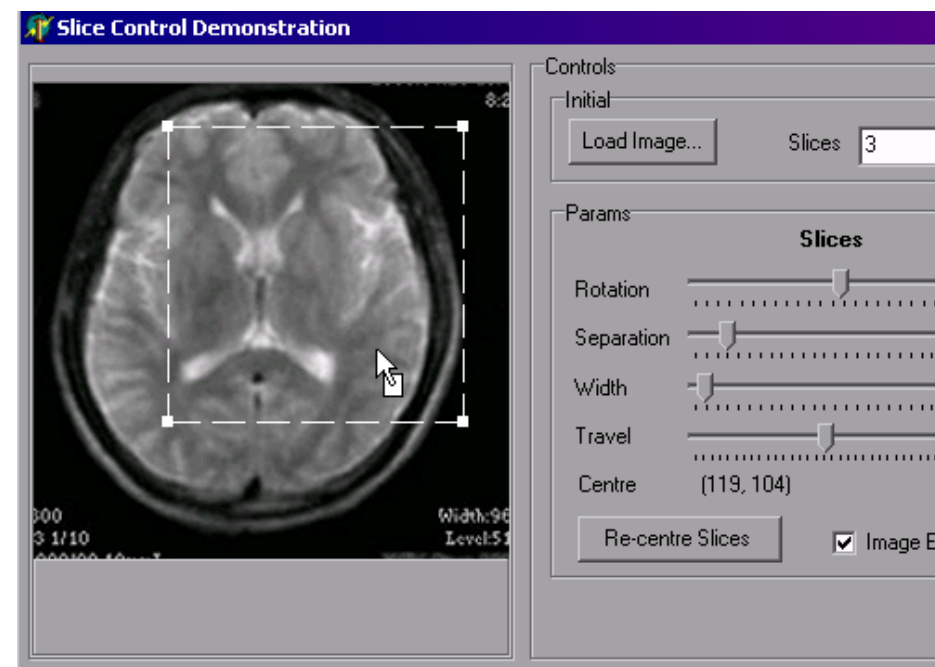
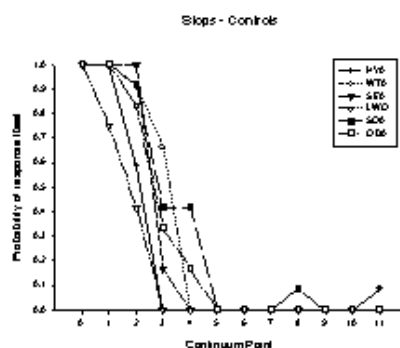


“stress impacts team performance ... by narrowing or weakening the team-level perspective required for effective team behavior.”

– Driskell et al, 1999



(not just 'pop' science!)





**High-performing teams are
hugely effective**

Optimise for the team



**Match the team
responsibility to the
cognitive load that
the team can handle**



Real-world Team Topologies



DevOps Team Topologies

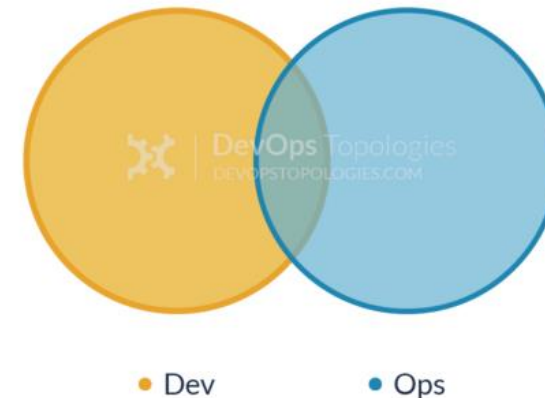
Having seen what makes the anti-types bad, we can look at some topologies in which DevOps can be made to work.

Topologies 1 2 3 4 5 6 7 8 9

Type 1: Smooth Collaboration

This is the 'promised land' of DevOps: smooth collaboration between Dev teams and Ops teams, each specialising where needed, but also sharing where needed. There are likely many separate Dev teams, each working on a separate or semi-separate product stack.

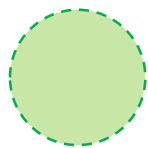
My sense is that the Type 1 Smooth Collaboration model needs quite substantial organisational change to establish it, and a good degree of competence higher up in the technical management team. Dev and Ops must have a clearly expressed and demonstrably effective shared goal ('Delivering Reliable, Frequent Changes', or whatever). Ops folk must be comfortable



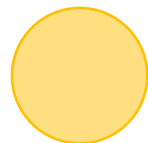
Type 1 suitability: an organisation with strong technical leadership.

Potential effectiveness: **HIGH**

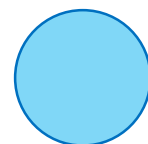
DevOpsTopologies.com



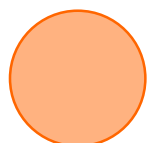
DevOps activity



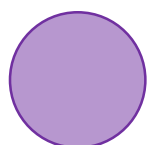
Development & Testing



IT Operations / Web Operations



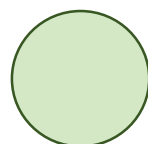
Database / DBA



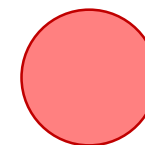
SRE



Component



Supporting (Tooling / Platform / Build)



Anti-Type



(Can you spot an important team type that is missing?)

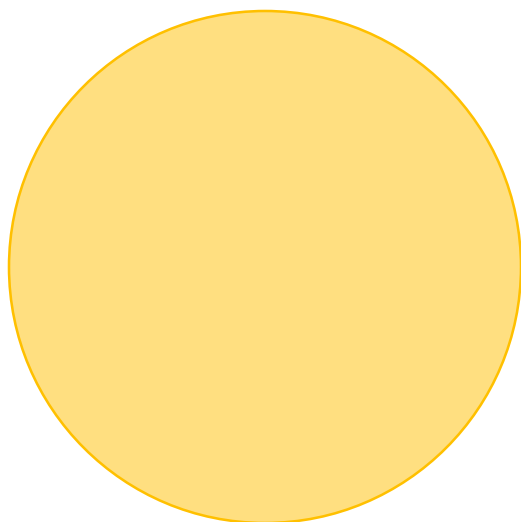


Anti-Types

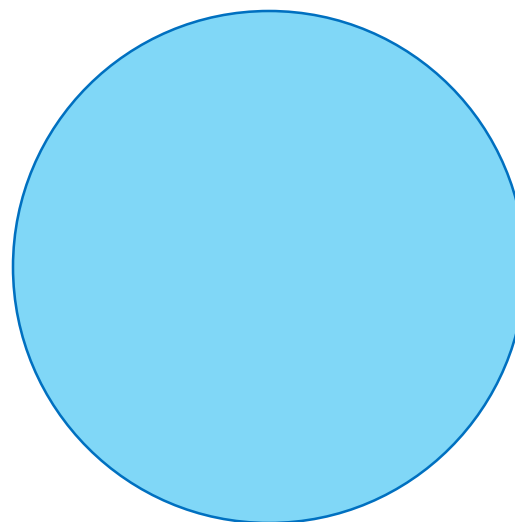


Anti-Type A – Separate Silos

devopstopologies.com



Dev

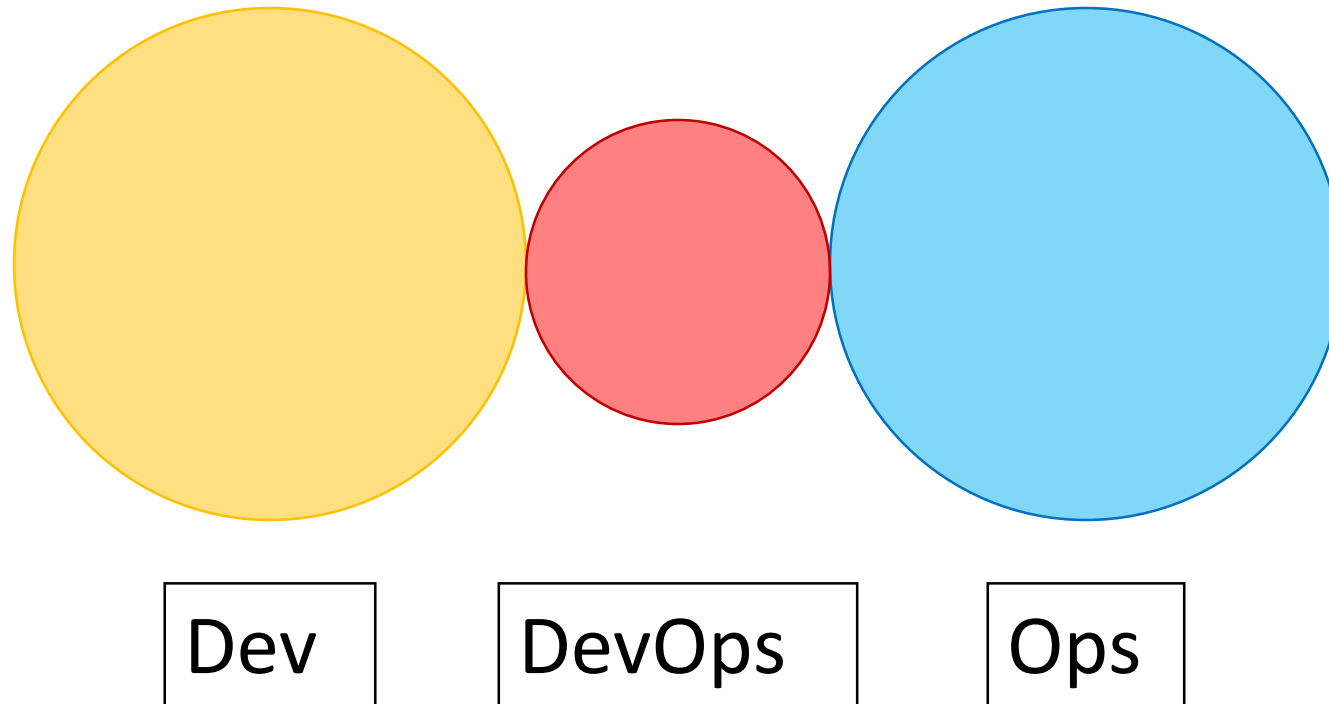


Ops



Anti-Type B – Separate DevOps Silo

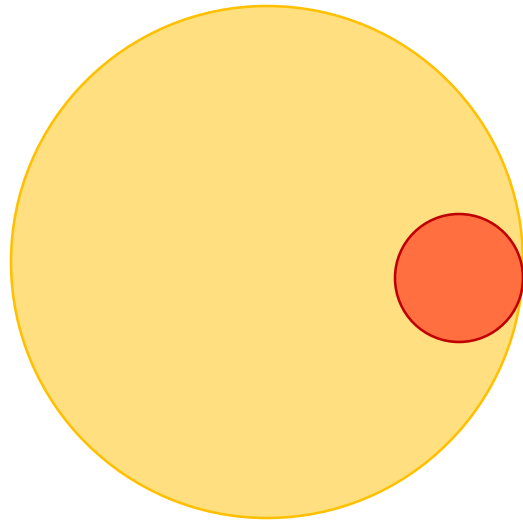
devopstopologies.com





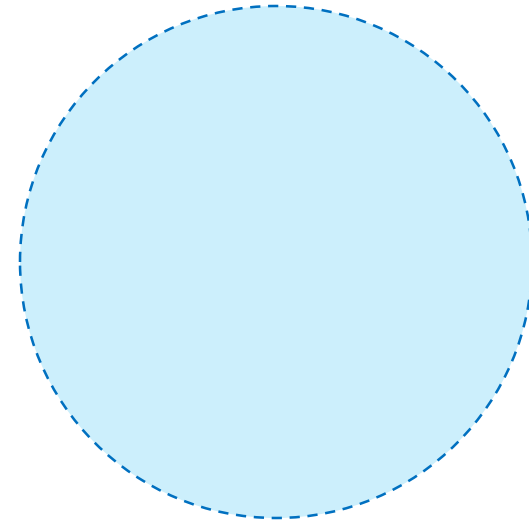
Anti-Type C – “We Don’t Need Ops”

devopstopologies.com



Dev

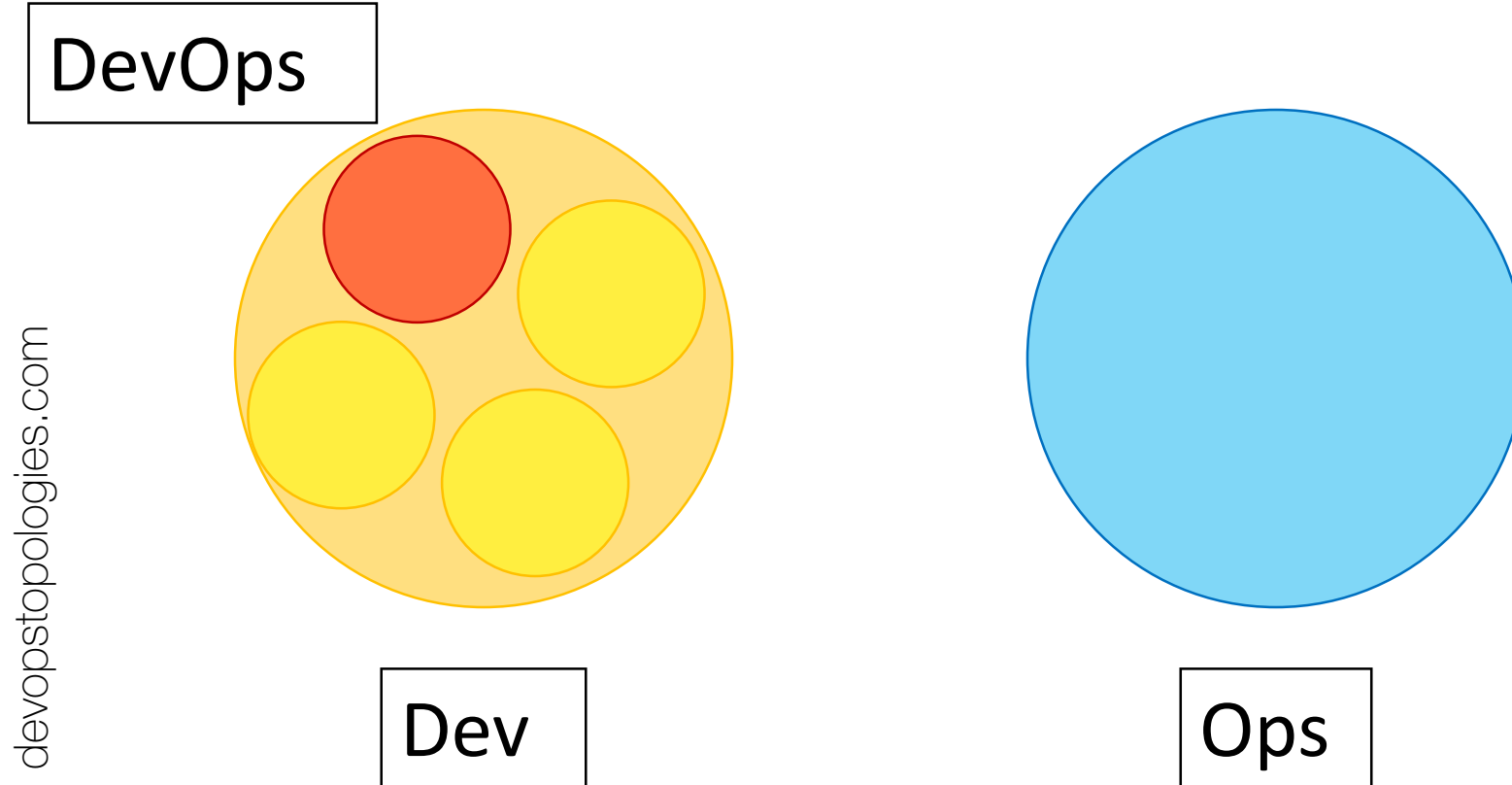
DevOps



Ops



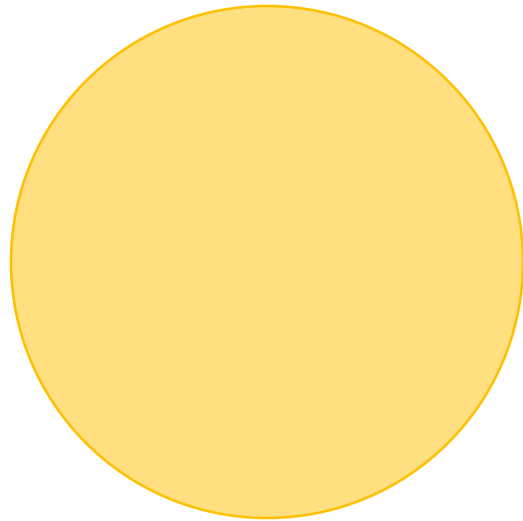
Anti-Type D – ‘DevOps’ as another Dev team



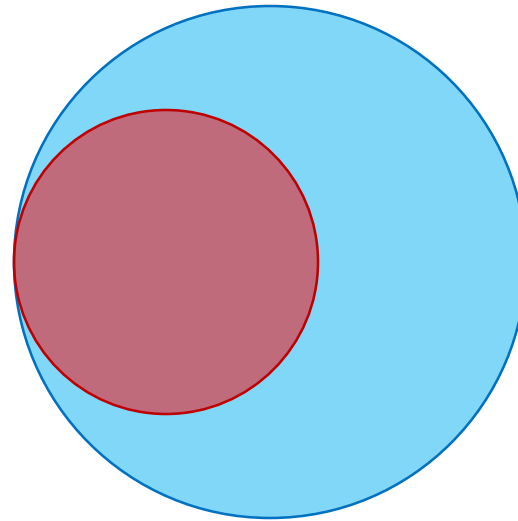


Anti-Type E – DevOps as new SysAdmin team

devopstopologies.com



Dev

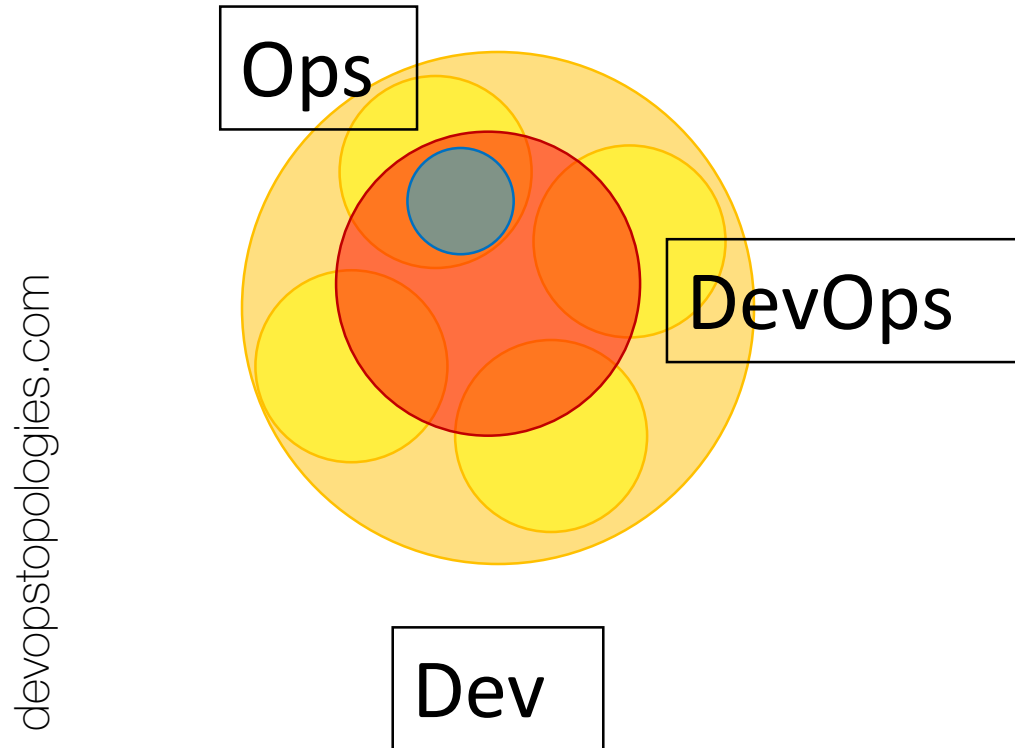


DevOps

Ops



Anti-Type F – Ops embedded in a Dev Team

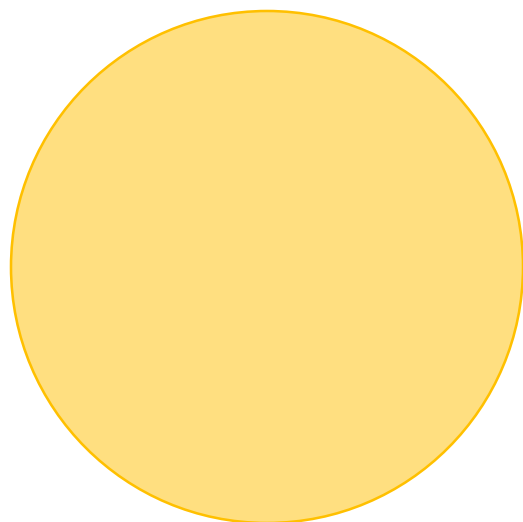


HT: Matt Franz (@seclectech)

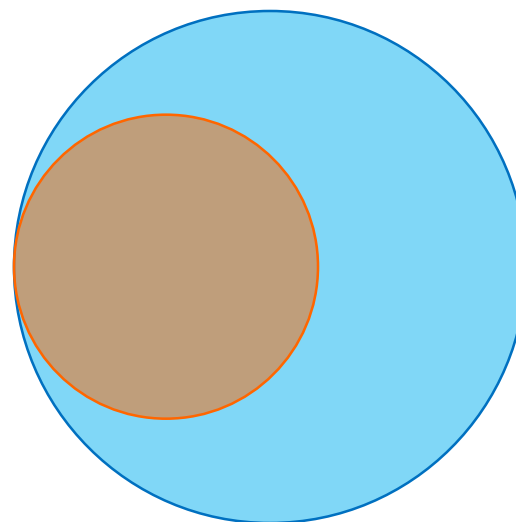


Anti-Type G – Dev-DBA gap!

devopstopologies.com



Dev



DBA

Ops

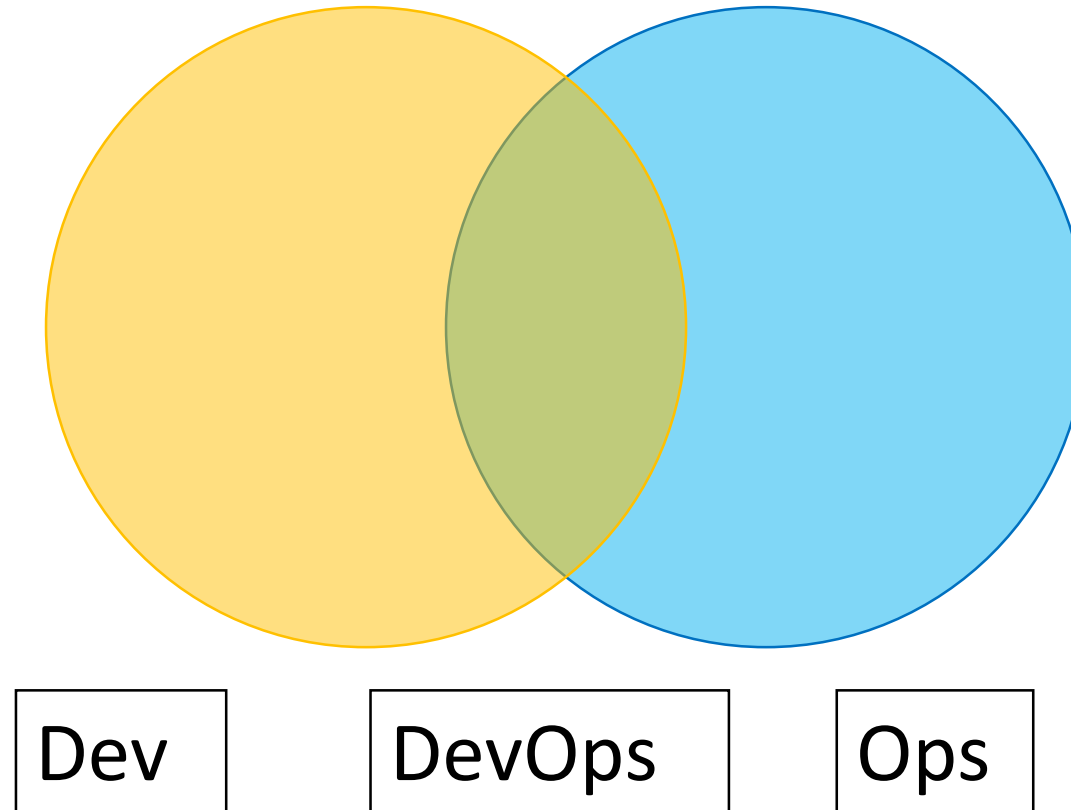


Types



Type 1 – Smooth Collaboration

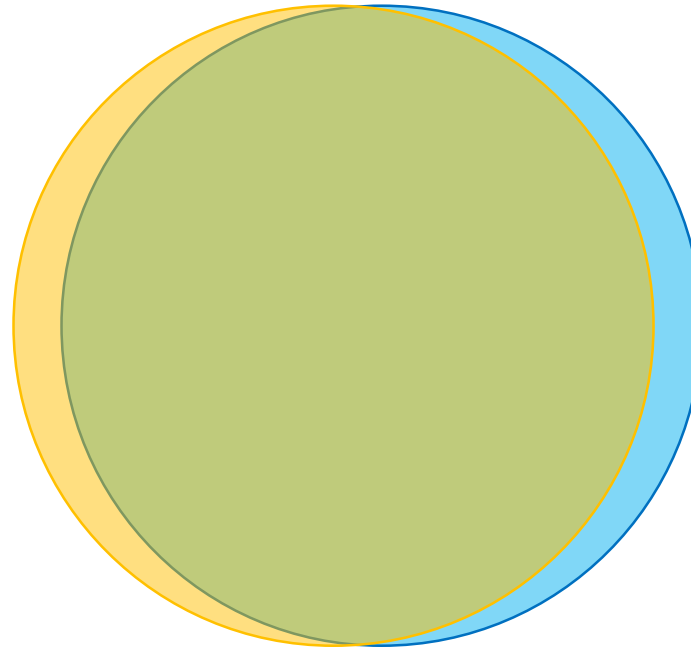
devopstopologies.com





Type 2 – Fully Embedded

devopstopologies.com



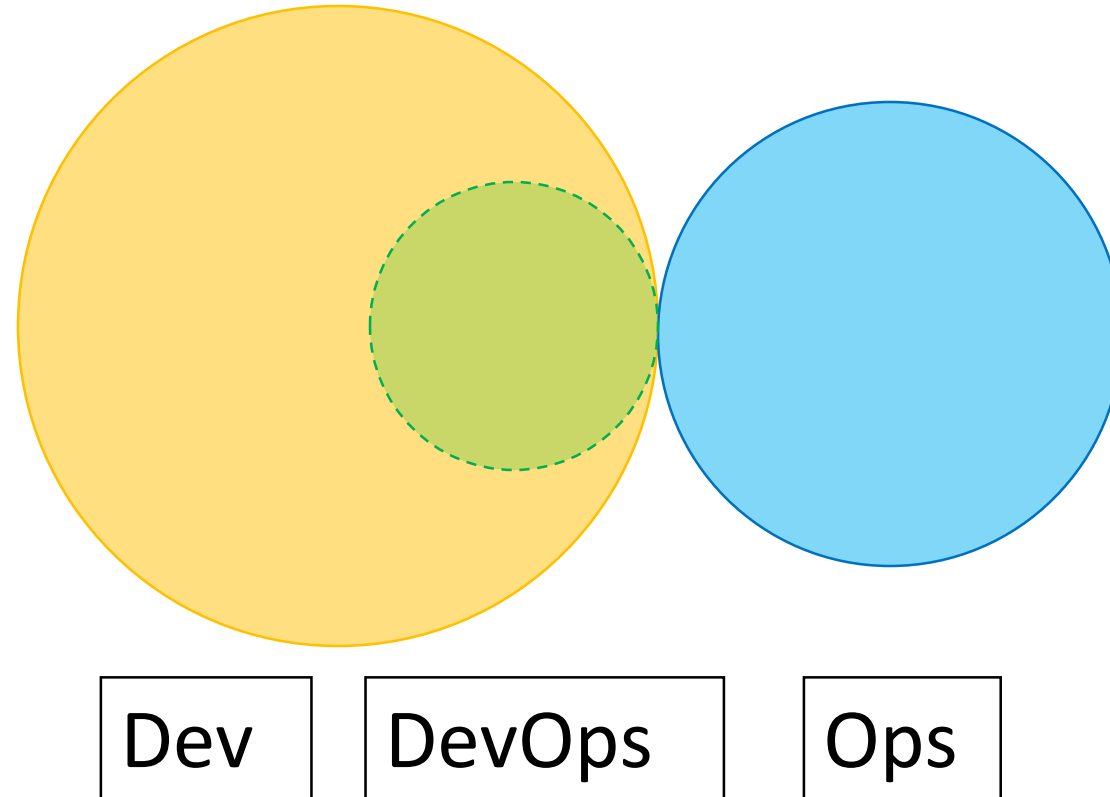
Dev

Ops



Type 3 – Infrastructure-as-a-Service

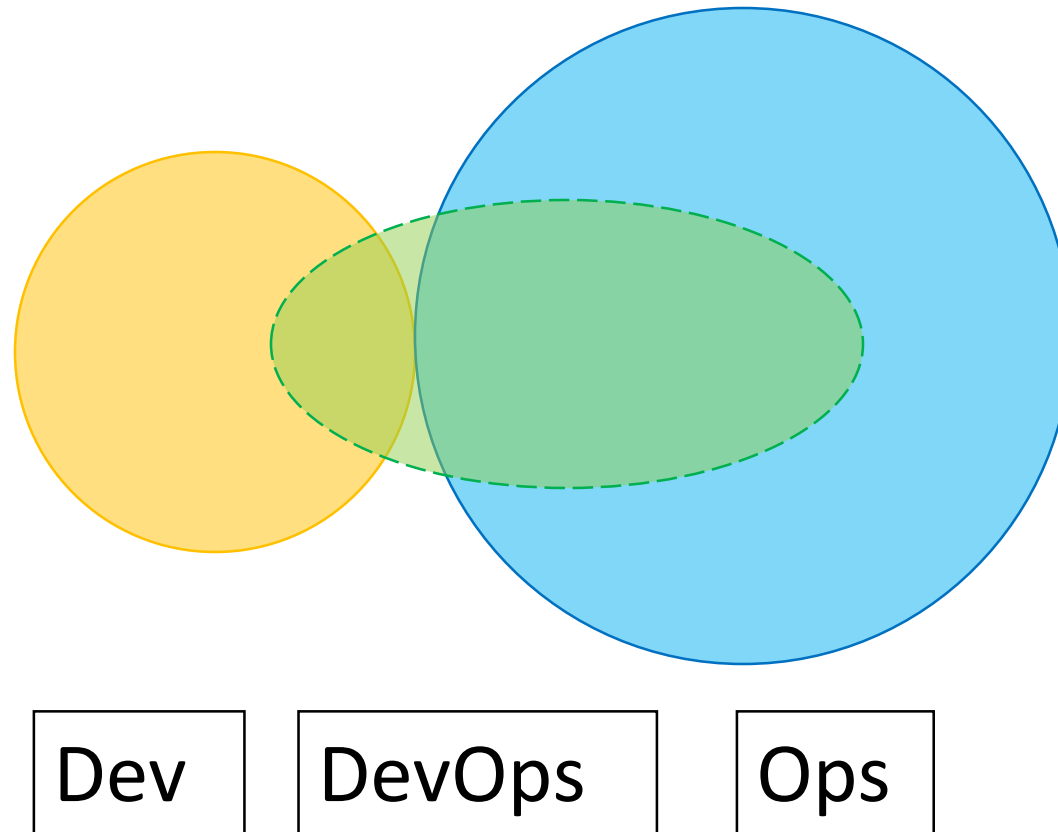
devopstopologies.com





Type 4 – DevOps-as-a-Service

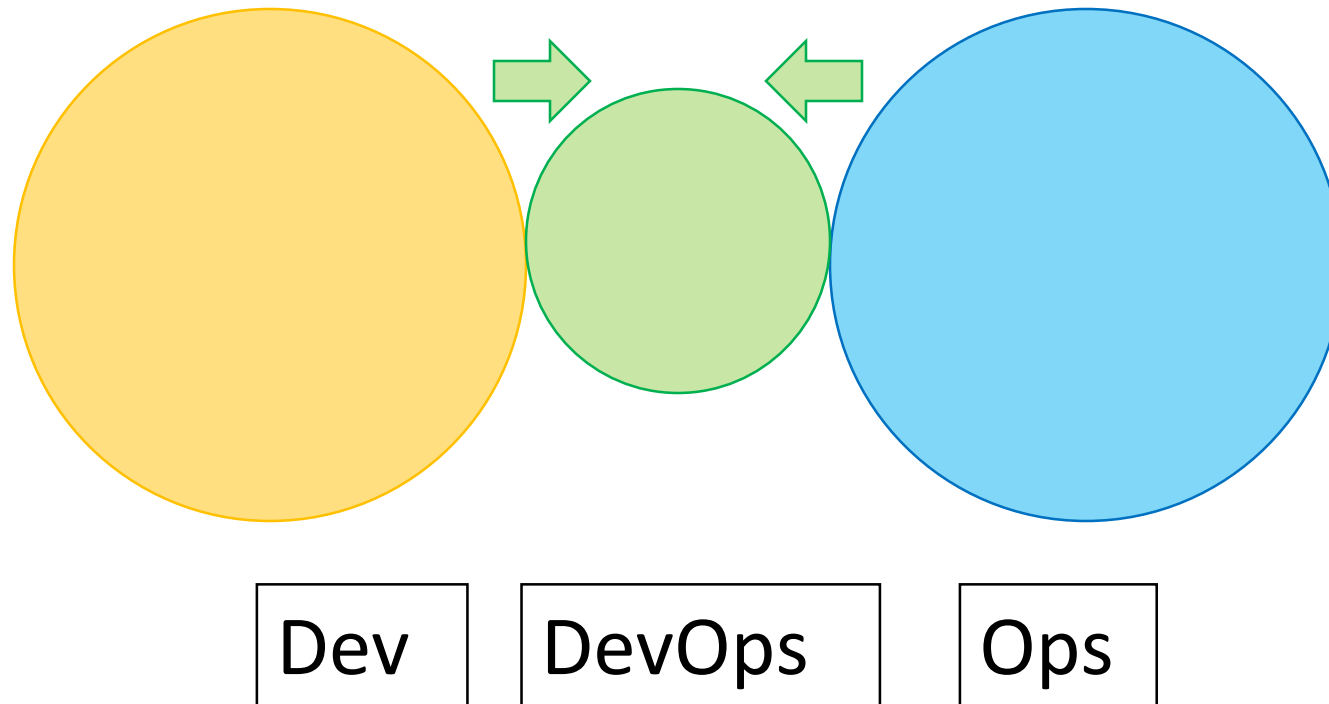
devopstopologies.com





Type 5 – Temporary DevOps Team

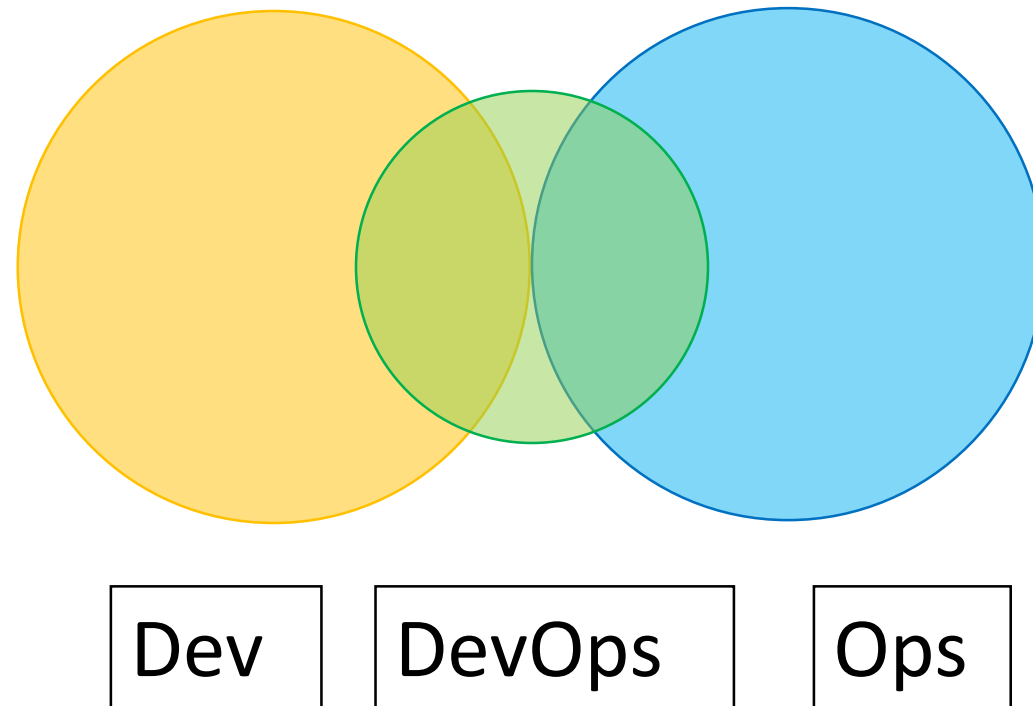
devopstopologies.com





Type 6 – ‘Facilitating’ DevOps Team

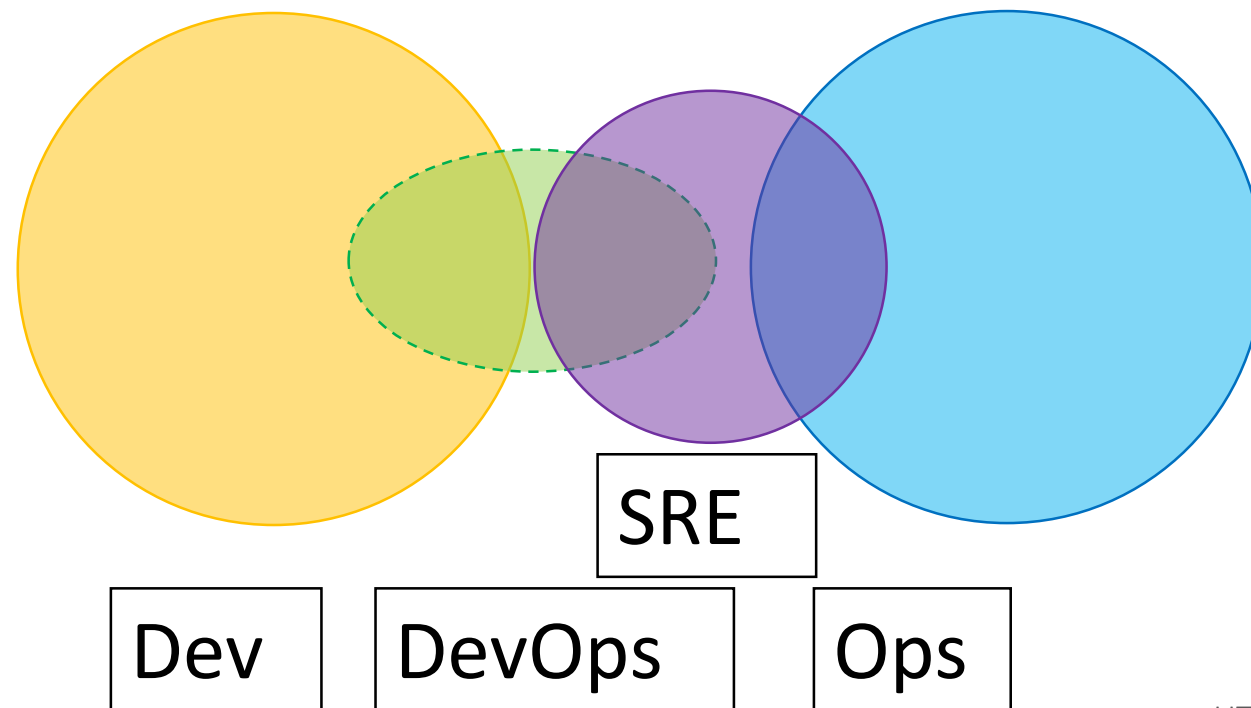
devopstopologies.com





Type 7 – SRE Team (Google)

devopstopologies.com

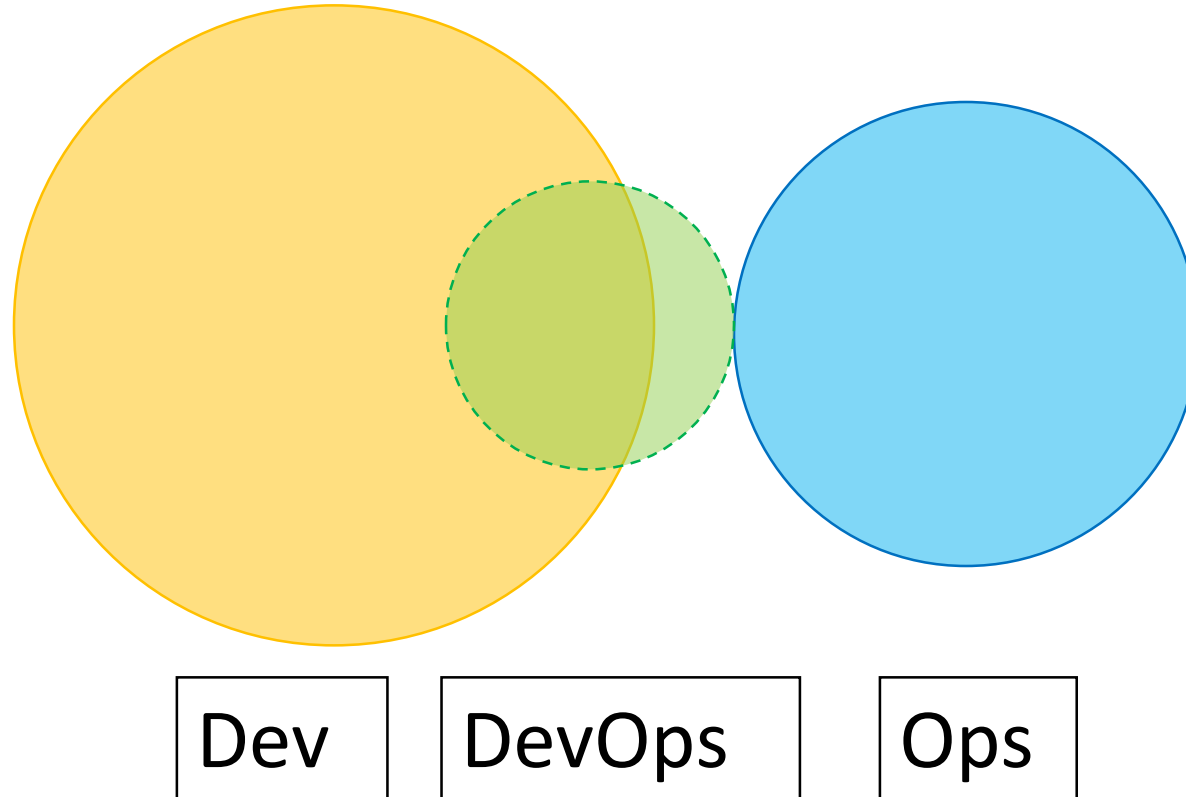


HT: @kwdhinde



Type 8 – ‘Just run my Containers’

devopstopologies.com

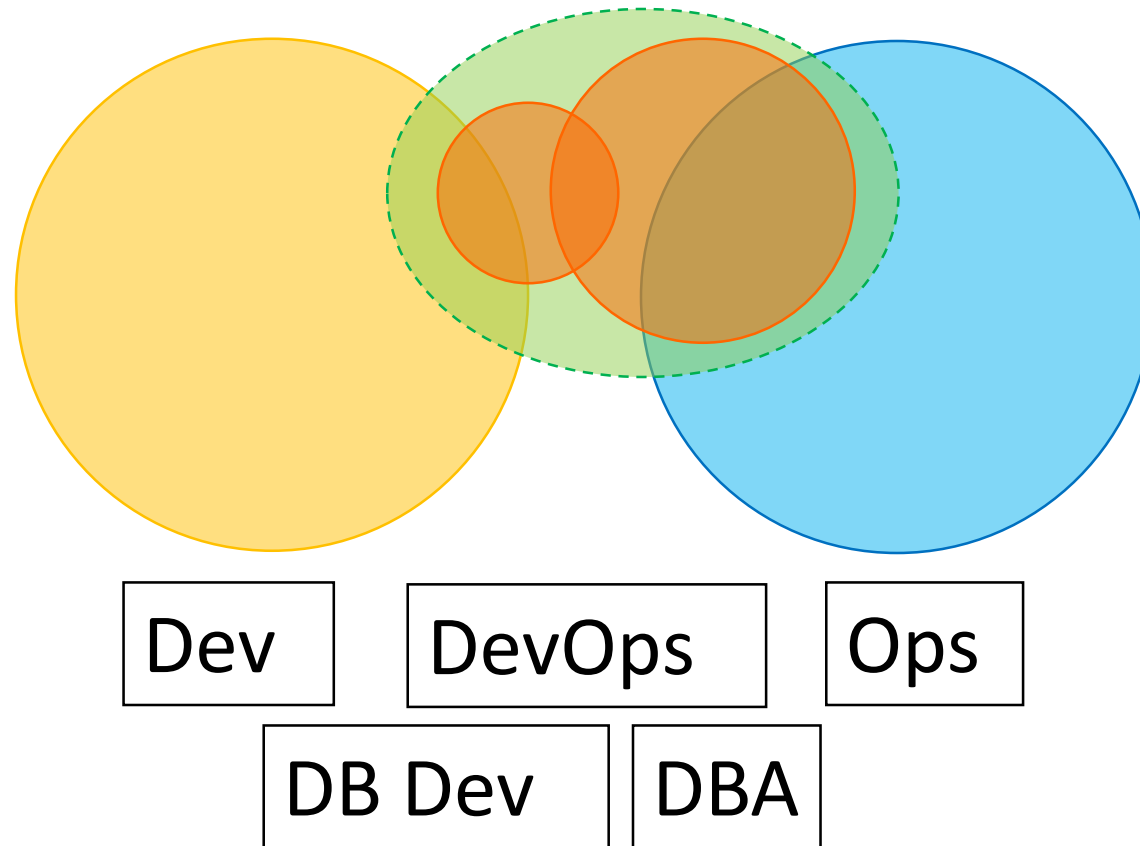


HT: @jascbu



Type 9 – DB capability in Dev

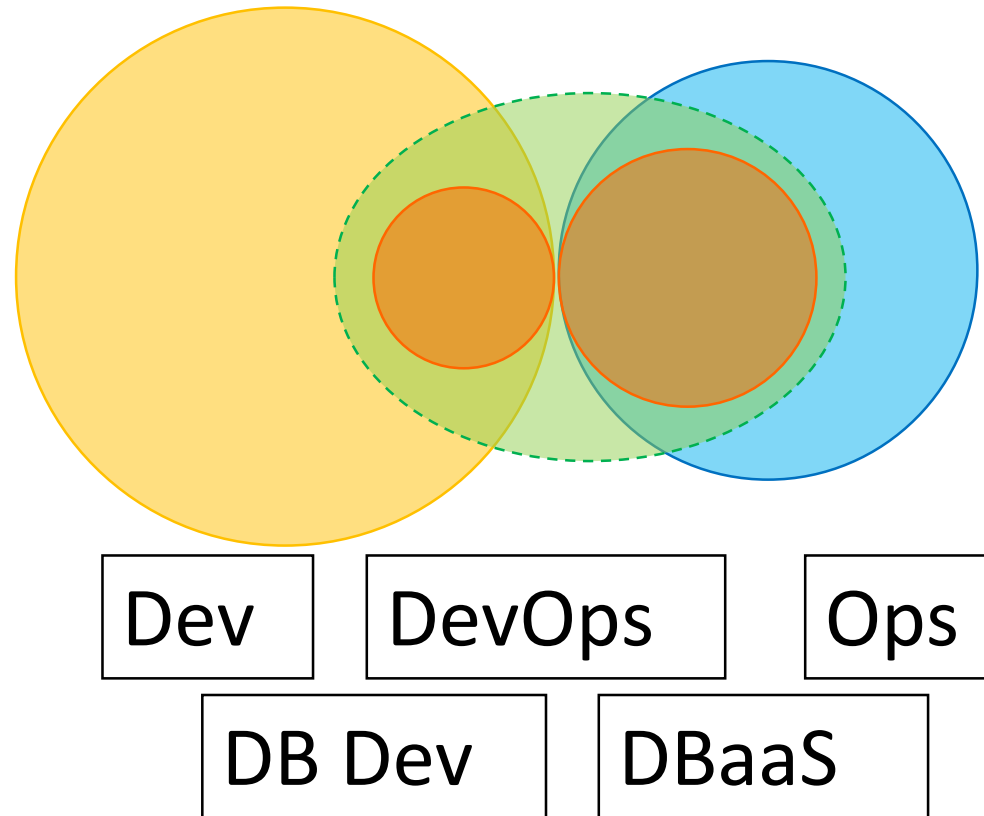
devopstopologies.com





Type 10 – DB as a Service

devopstopologies.com





There is no single 'right' team topology, but several 'bad' topologies for any one organisation

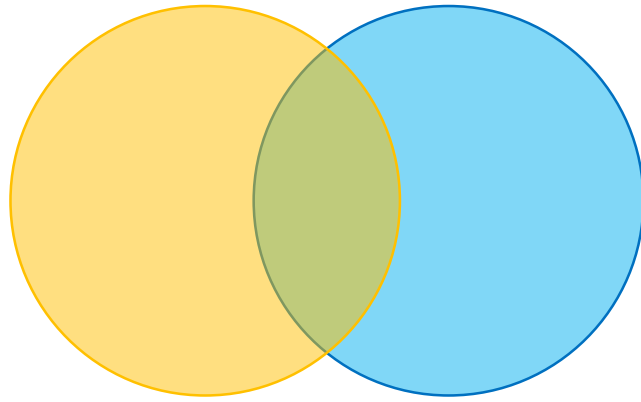


Guidelines for team design

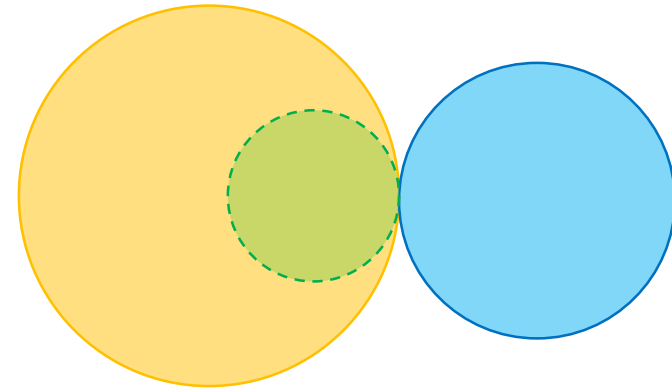


Collaboration vs X-as-a-Service

devopstopologies.com



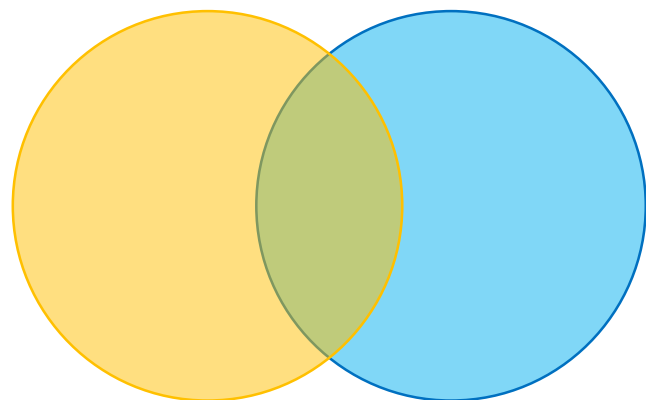
Collaboration



X-as-a-Service



Collaboration vs X-as-a-Service



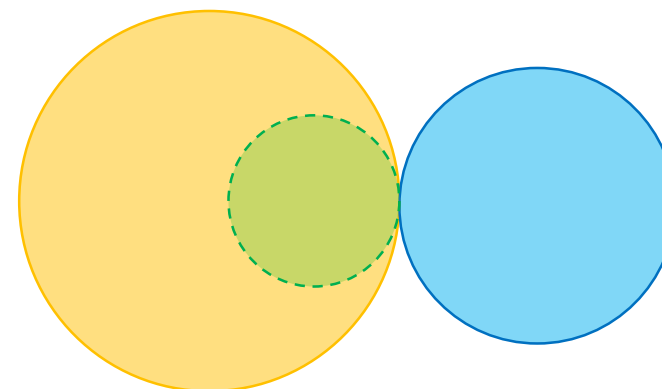
Collaboration



Rapid discovery
No hand-offs



Comms overheads?



X-as-a-Service



Ownership clarity
Less context needed

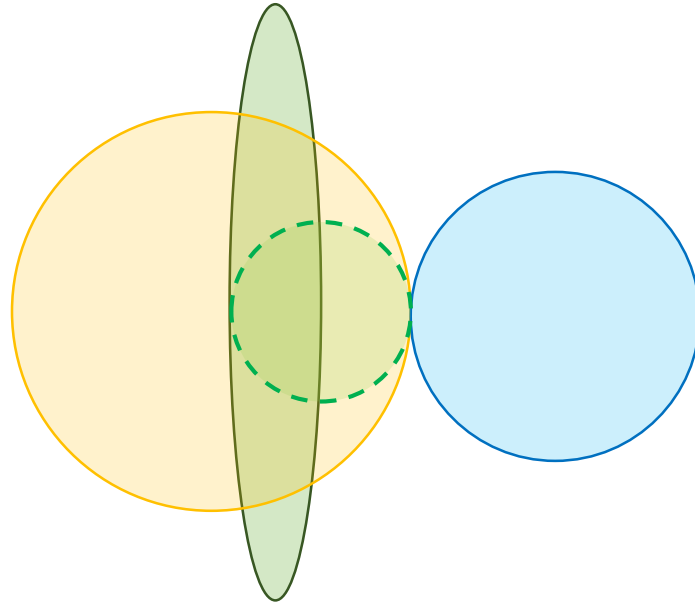


Slower innovation?

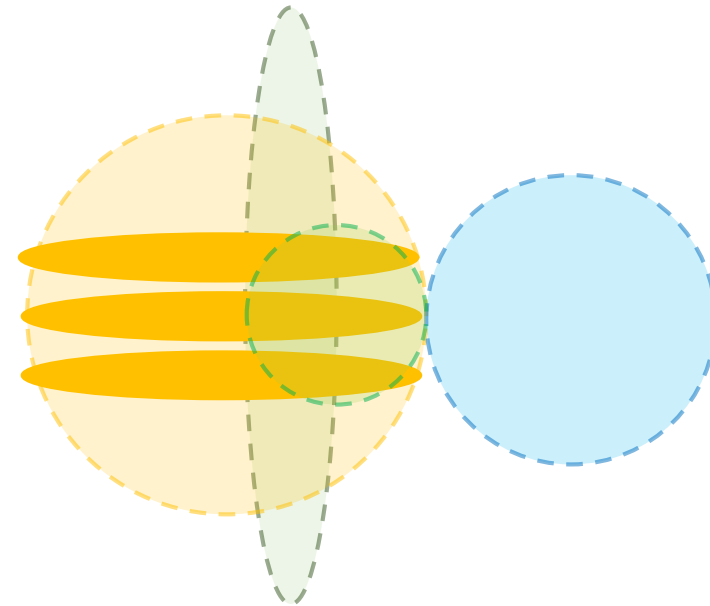


Supporting & Business Domain

devopstopologies.com



Supporting

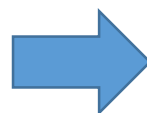
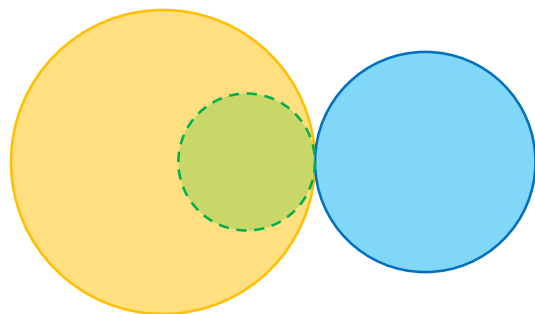


Business Domain

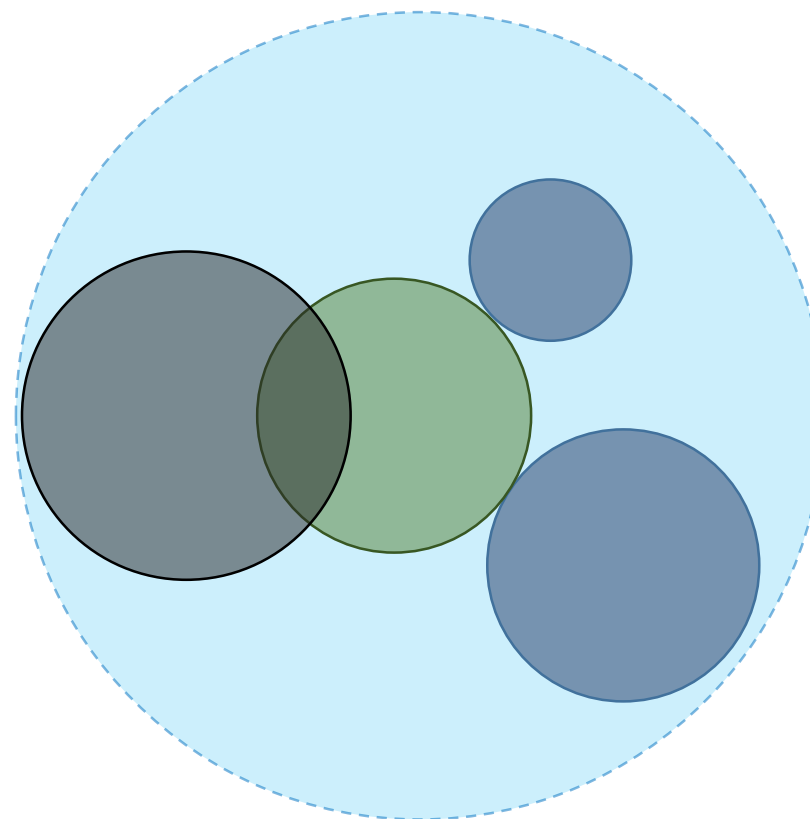


Inner Topologies

devopstopologies.com



Within any group there may be internal collaborations AND other X-as-a-Service (XaaS) relationships



Collaboration

XaaS



Team types

devopstopologies.com



Product/Feature team



Platform / 'substrate' team



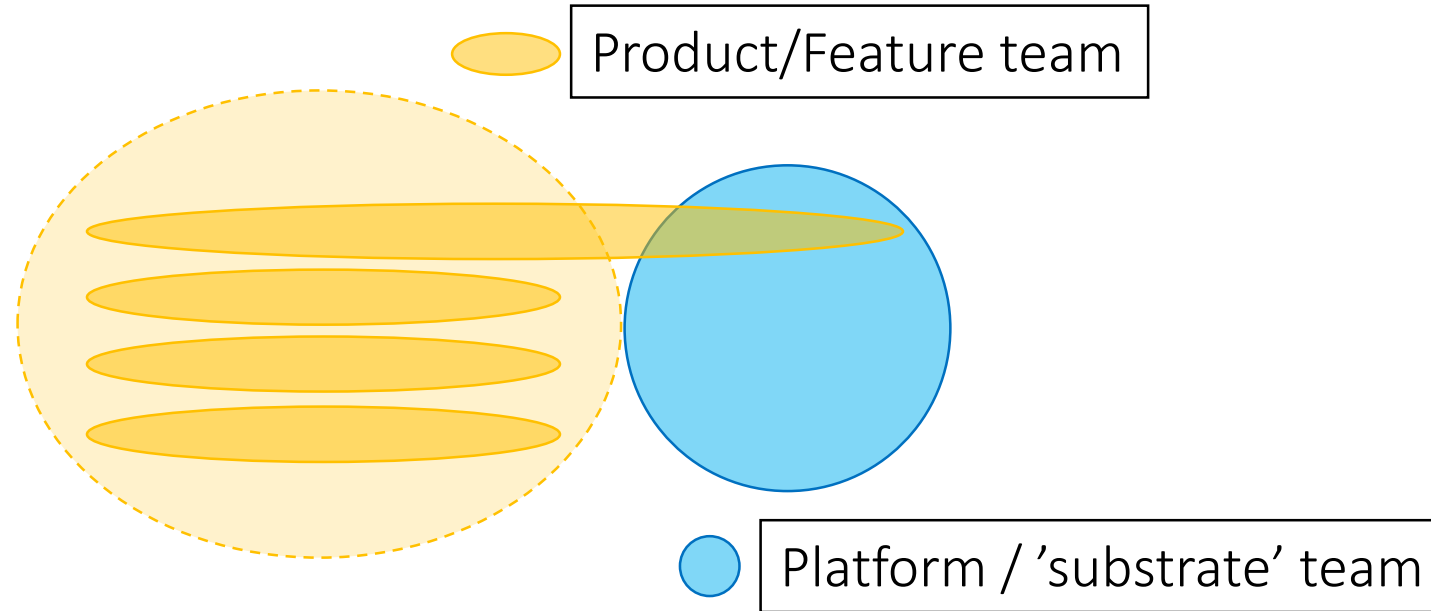
Component team



Supporting / 'productivity' team

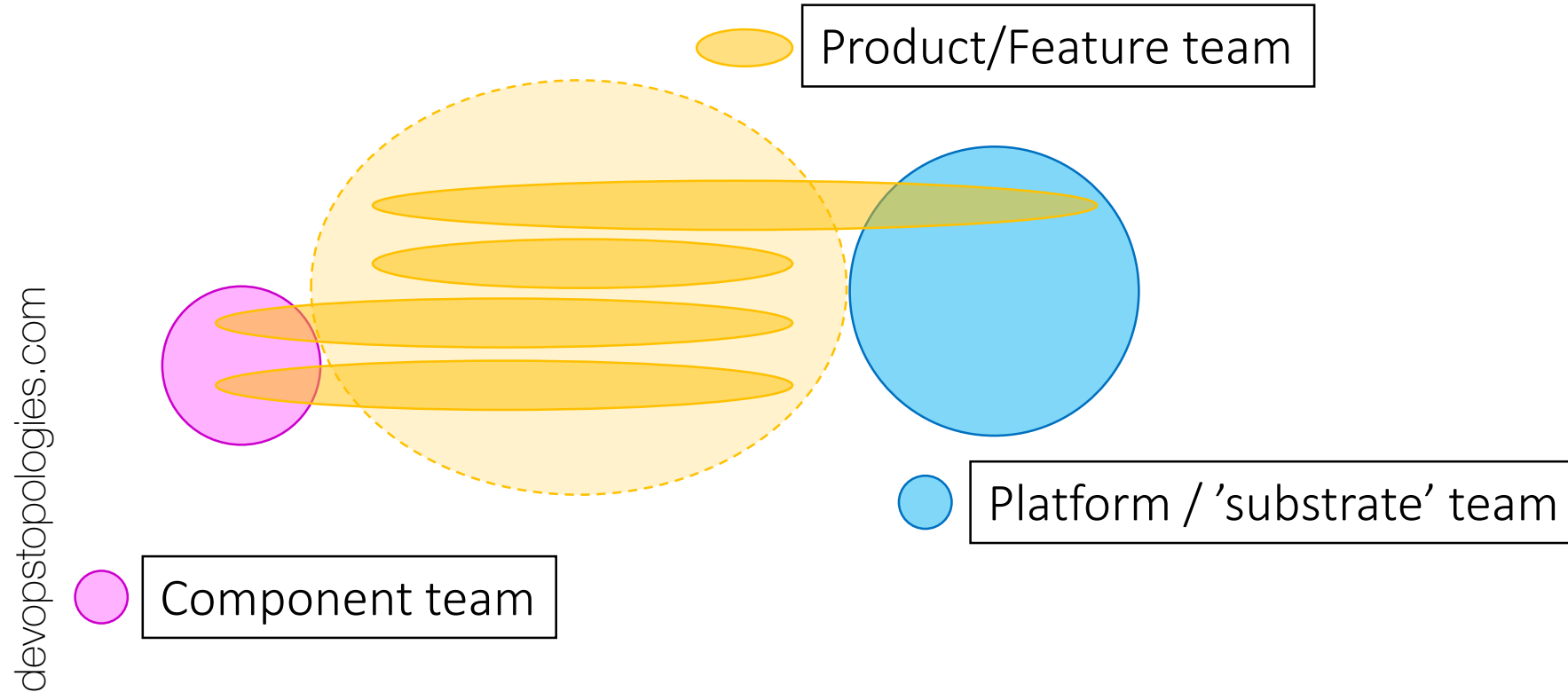


Team configuration



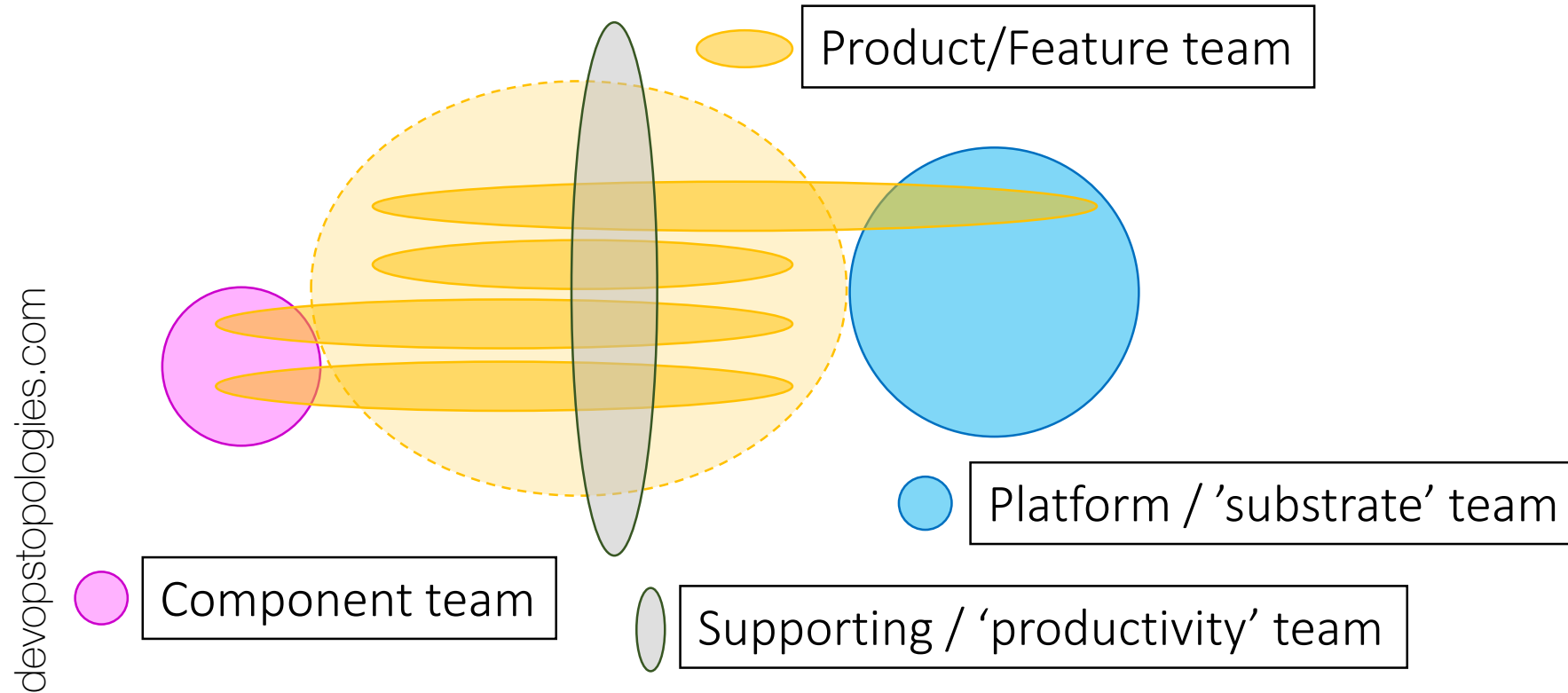


Team configuration



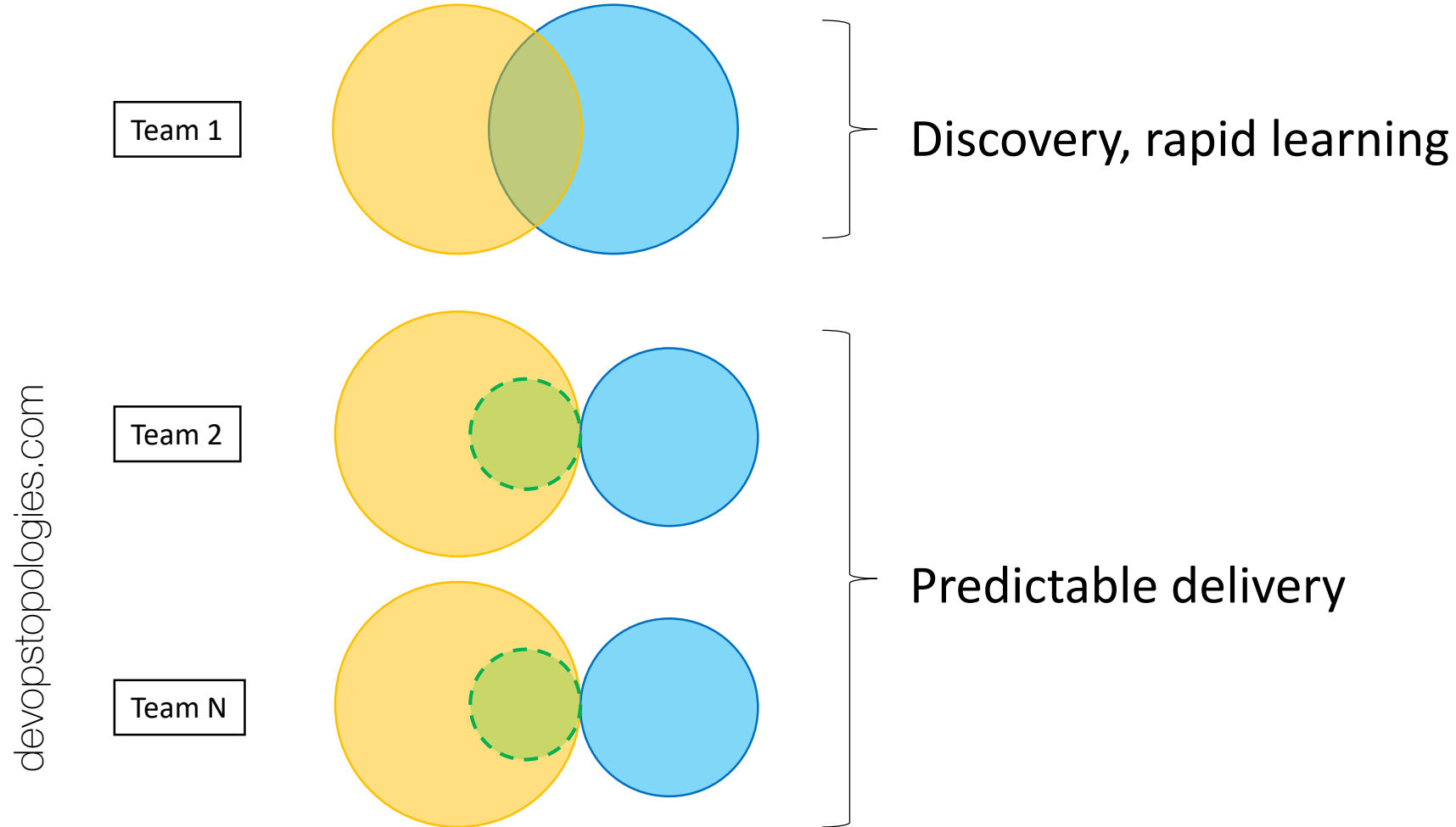


Team configuration





Discovery vs. Predictability

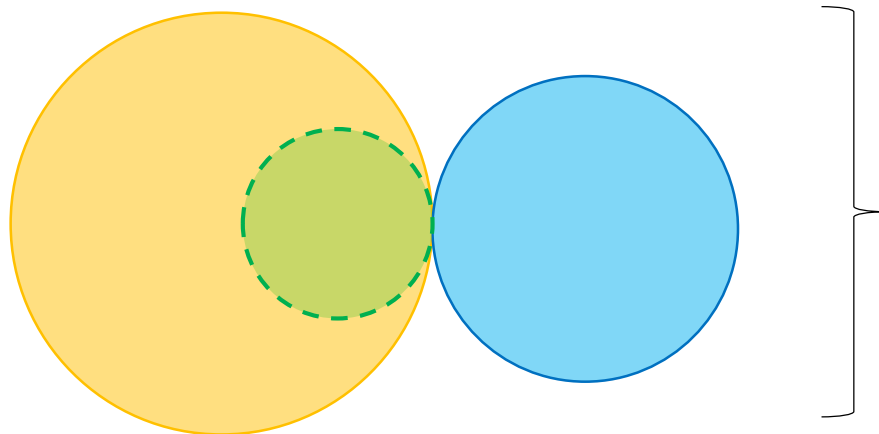




Established platform (PaaS)



devopstopologies.com



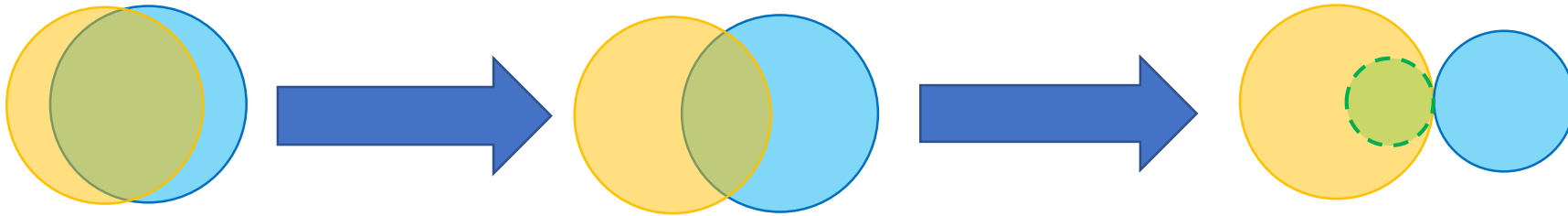
Predictable delivery



Evolution of team topologies

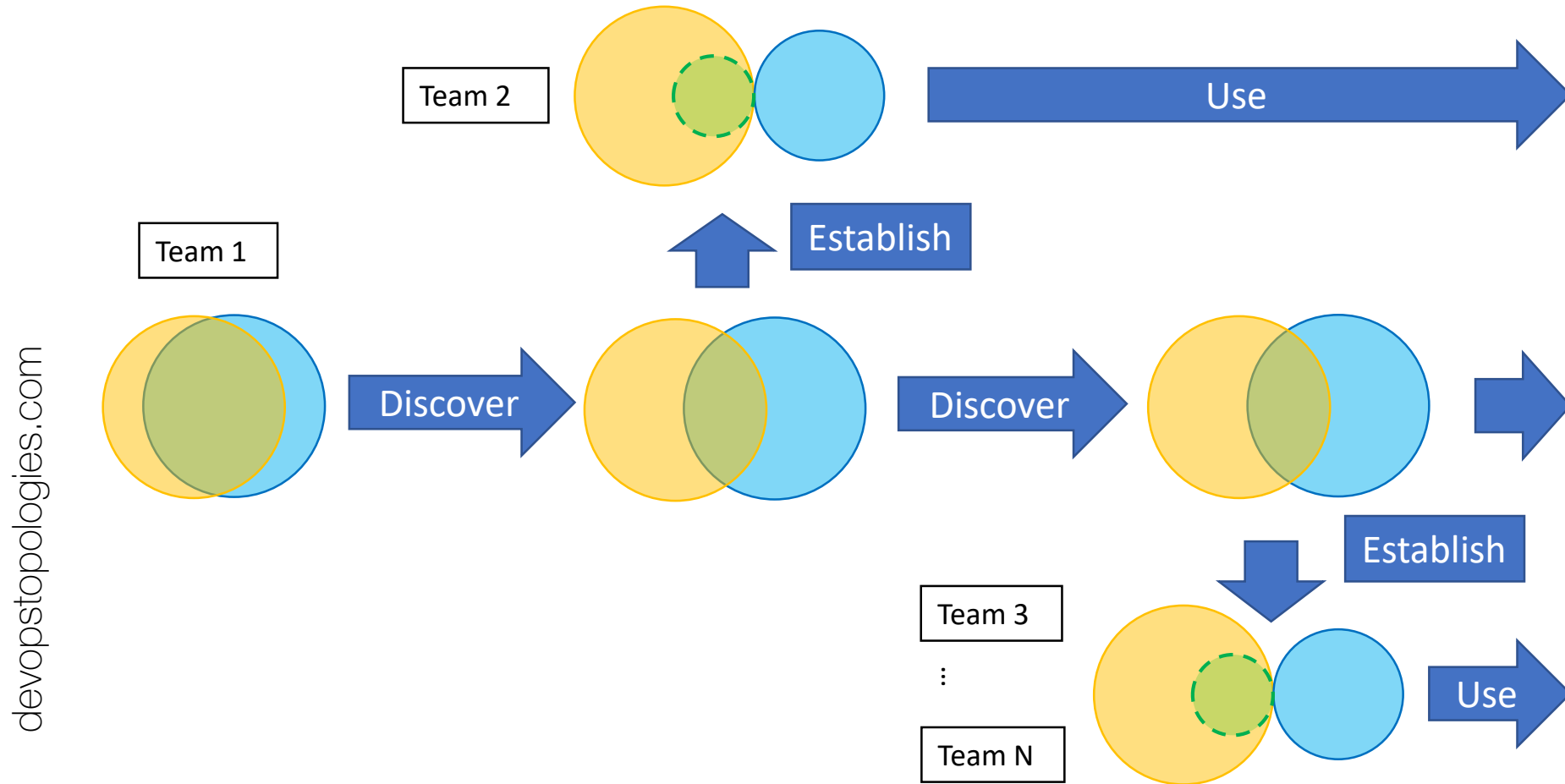
DISCOVER

ESTABLISH





Evolution of team topologies





**Evolve different team
topologies for different parts of
the organisation at different
times to match the team
purpose and context**

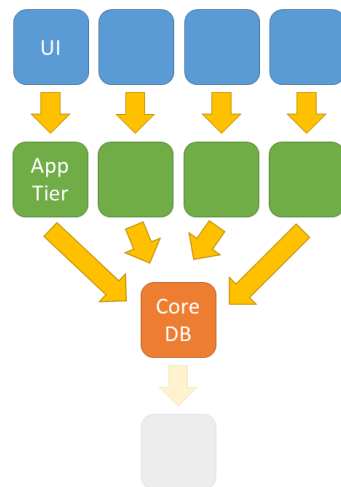
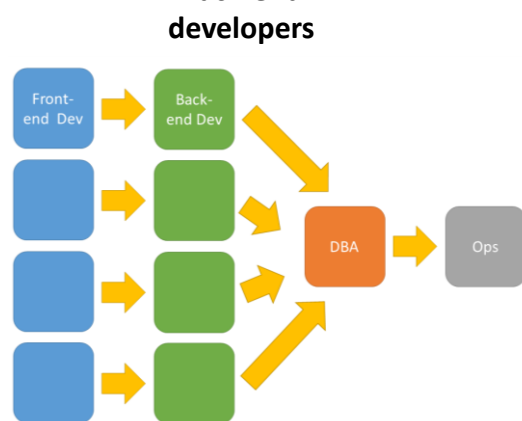


Summary



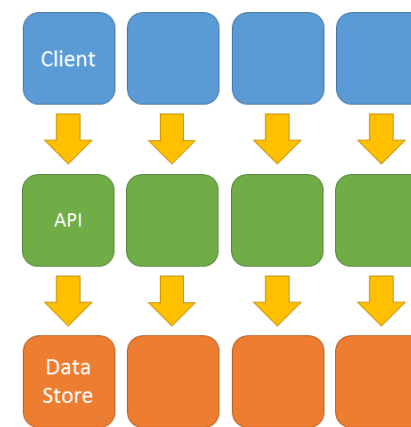
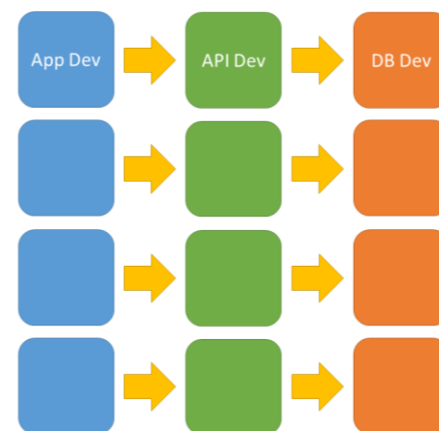
Front-end
developers

Back-end
developers



A

B



A

B



**Design the
organisation architecture
to produce the right software
architecture**



“stress impacts team performance ... by narrowing or weakening the team-level perspective required for effective team behavior.”


– Driskell et al, 1999

Group Dynamics: Theory, Research, and Practice 1999, Vol. 3, No. 4, 291-302



**Match the team responsibility
to the cognitive load that the
team can handle**



 DevOps Topologies

Anti-TypesTeam Topologies

DevOps Team Topologies

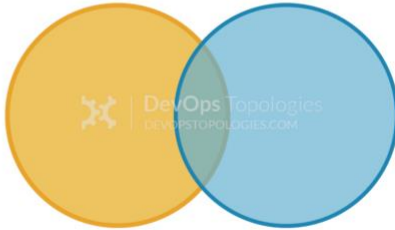
Having seen what makes the anti-types bad, we can look at some topologies in which DevOps can be made to work.

Topologies123456789

Type 1: Smooth Collaboration


This is the 'promised land' of DevOps: smooth collaboration between Dev teams and Ops teams, each specialising where needed, but also sharing where needed. There are likely many separate Dev teams, each working on a separate or semi-separate product stack.

My sense is that the Type 1 Smooth Collaboration model needs quite substantial organisational change to establish it, and a good degree of competence higher up in the technical management team. Dev and Ops must have a clearly expressed and demonstrably effective shared goal ('Delivering Reliable, Frequent Changes', or whatever). Ops folk must be comfortable



• Dev

• Ops

 **Type 1 suitability:** an organisation with strong technical leadership.
Potential effectiveness: **HIGH**

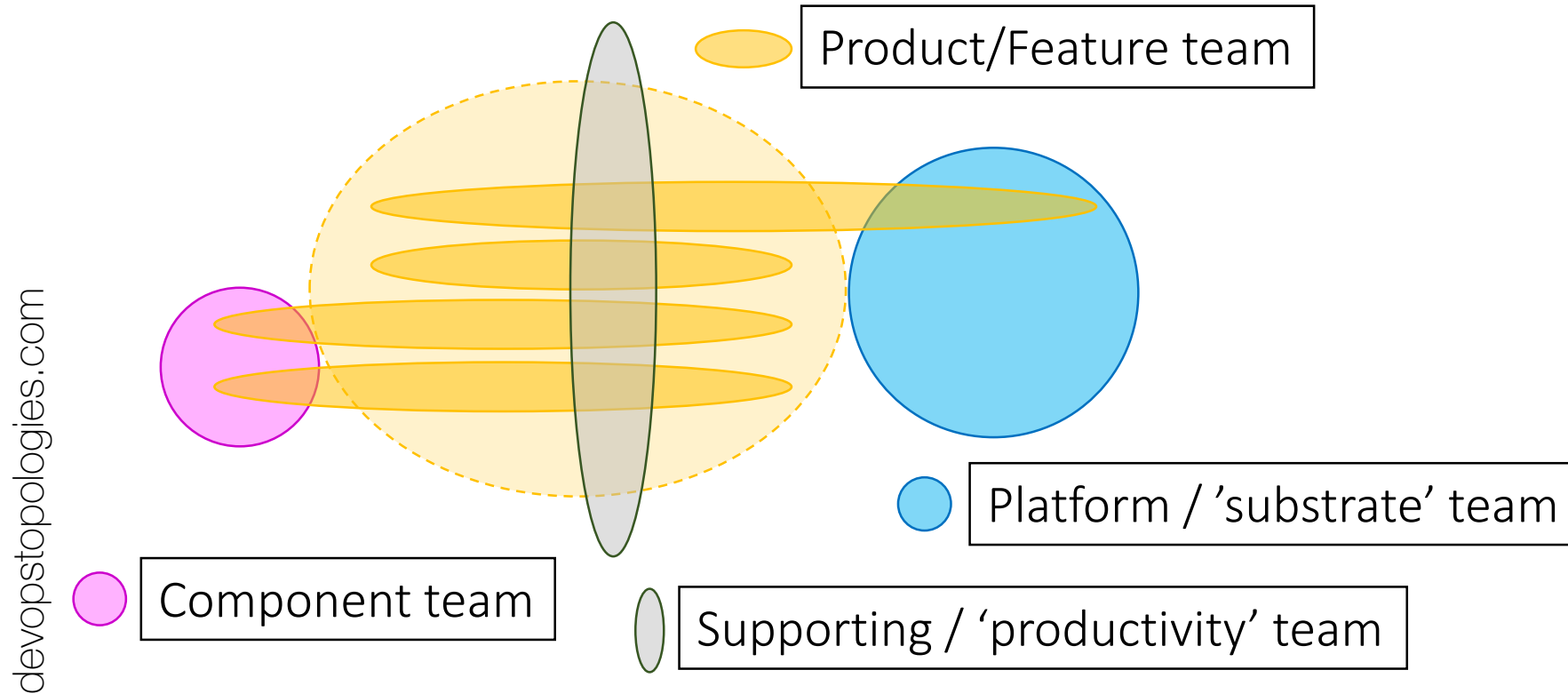
DevOpsTopologies.com



There is no single 'right' team topology, but several 'bad' topologies for any one organisation

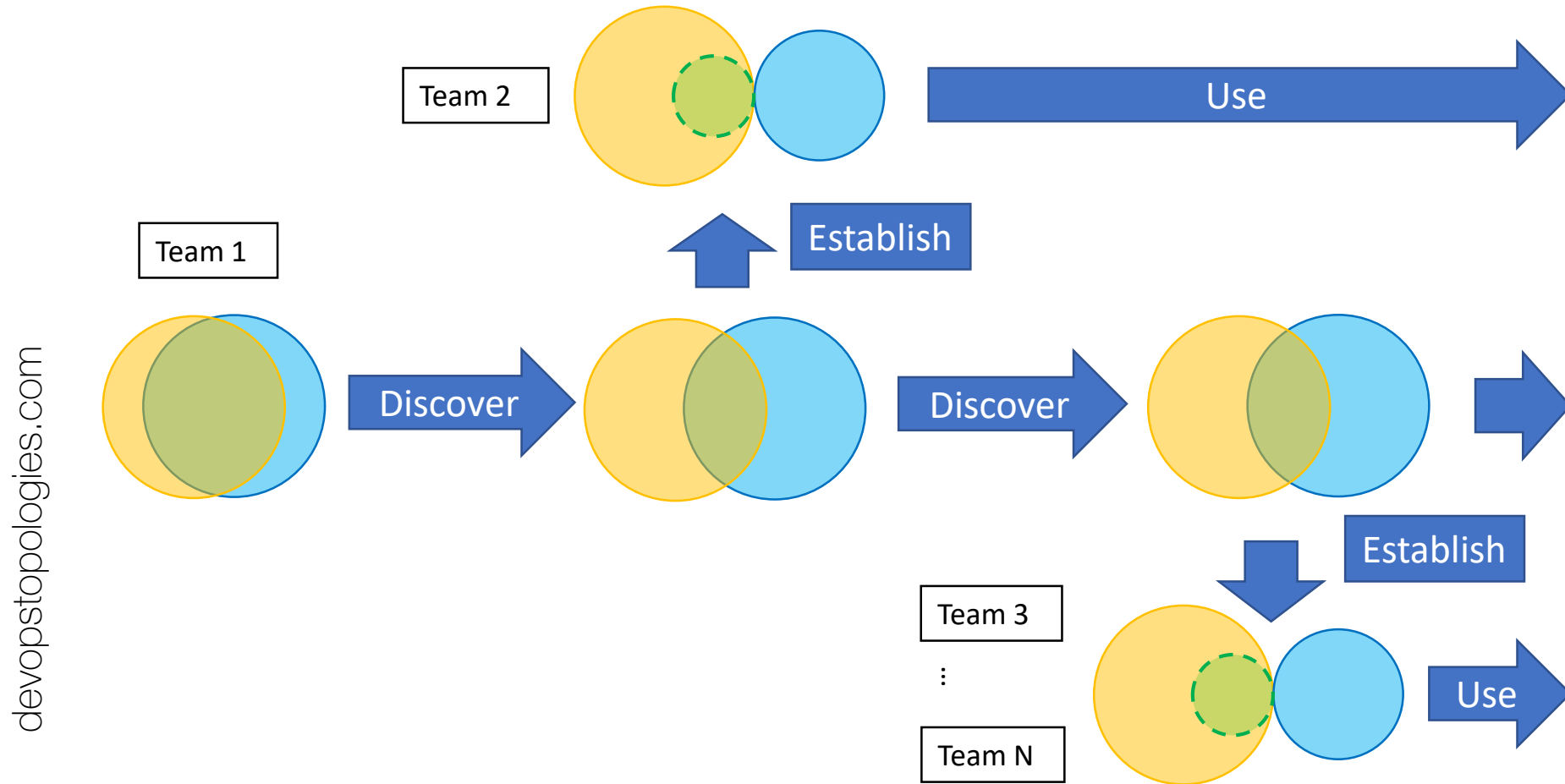


Team configuration





Evolution of team topologies





**Evolve different team
topologies for different parts of
the organisation at different
times to match the team
purpose and context**



Caution



**Team topologies
alone will not
produce effective
software systems**



Also needed:
culture, good engineering,
sane funding models,
clarity of business vision



Safer, more rapid
changes to
software systems
(Business Agility)



SKELTON THATCHER
CONSULTING

skeltonthatcher.com

Use code
"DOES17" for
25% discount

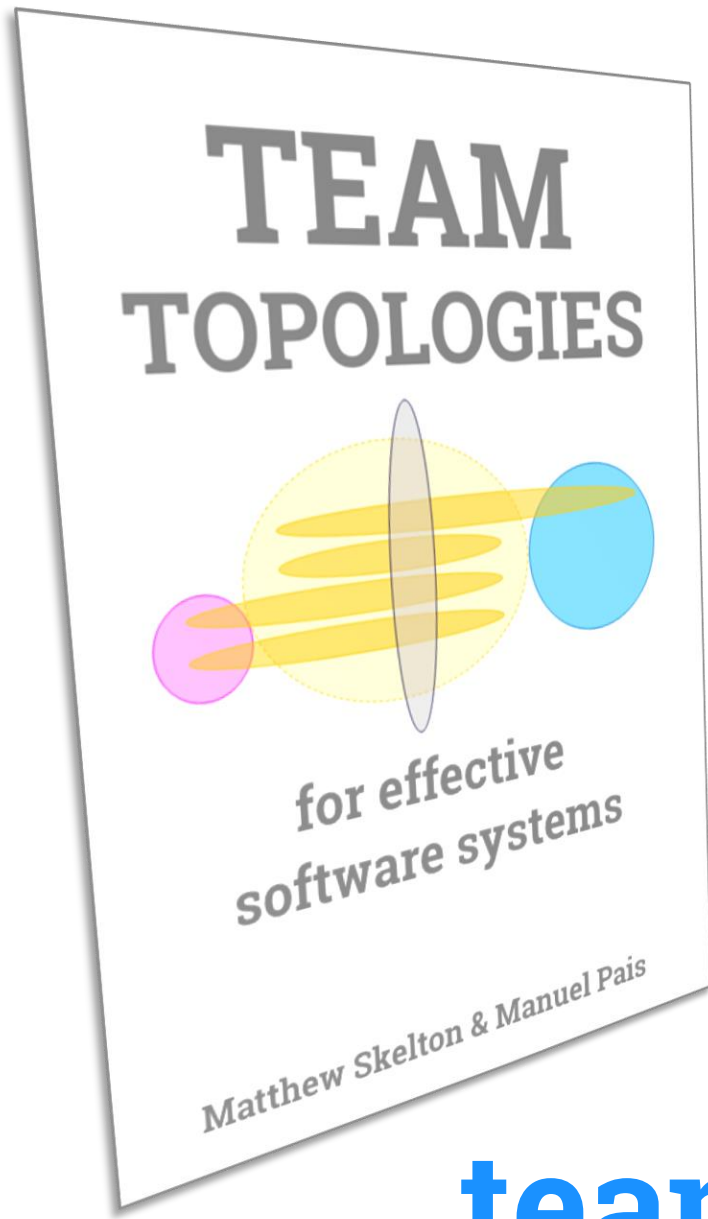


SKELTON THATCHER
EFFECTIVE SOFTWARE OPERATIONS

Organisation design for effective software systems - Tutorial / Workshop -
Sept 2017

📅 September 27th, 2017 📍 London, UK

<https://ti.to/skelton-thatcher-consulting/organisation-design-workshop-sept-2017>



Upcoming book:

*Team Topologies for
effective software systems*

by Matthew Skelton & Manuel Pais

teamtopologies.com



thank you



Matthew Skelton & Manuel Pais
@SkeltonThatcher



skeltonthatcher.com