

GOTopia

FEBRUARY 2021

Platform Engineering As A (Community) Service

Nicki Watt - CTO

@techiewatt



| About Me



Nicki Watt
Chief Technical Officer

@Techiewatt

OpenCredo

A TRIFORK COMPANY

- Data & Graph Systems
- Cloud Native Architectures
- Platform Engineering



Our Observation

In the push to adopt innovative technology platforms, we have lost sight of the human innovators & communities actually trying to use it

| Agenda

01. What Is Platform Engineering?

02. Target Communities

03. Characteristics Of A Successful Platform Experience

04. How To “Platform Engineer” Well

05. Conclusion



1

What Is Platform Engineering?



| A Platform Is ...

- “A foundation of self-service APIs, tools, services, knowledge and support which are arranged as a compelling internal product. Autonomous delivery teams can make use of the platform to deliver product features at a higher pace, with reduced co-ordination”
- *Evan Bottcher*
- “A curated experience for engineers (the customers of the platform)”
- *Matthew Skelton (Team Topologies)*

A Platform Is Not ...

- Just a technology (Kubernetes alone does not a platform make!)
- Just a wrapper around a bunch of tech



Why Build A Platform?

- **Cloud Native Solutions** are a response to business pressures but can be complex
- **Platforms** help tame this complexity & achieve business goals more effectively



Why Build A Platform?

- **Cloud Native Solutions** are a response to business pressures but can be complex
- **Platforms** help tame this complexity & achieve business goals more effectively



- Improve Engineering Productivity
- Provide Consistency & Confidence
- Help Scale Teams

More Markets

24/7

Bigger!!

Faster!!



| Platform Engineering Is ...

A discipline which involves doing whatever it takes to build, maintain & provide a curated platform experience for the communities using it



2

Who Are These Communities?



| Communities

- Traditional Engineering Teams

← **Most Obvious**

| Communities

- **Traditional Engineering Teams**
 - IaC, CI/CD, DevOps & SysOps help

← **Most Obvious**

| Communities

- **Traditional Engineering Teams**
 - IaC, CI/CD, DevOps & SysOps help
- **Data Analysts & Scientists**

← **Most Obvious**

← **Less Obvious**

| Communities

- **Traditional Engineering Teams**

- IaC, CI/CD, DevOps & SysOps help

← **Most Obvious**

- **Data Analysts & Scientists**

- MLOps, Specialised HW/SW (GPUs etc),
Version Tracking (Code,Data,Parameters)

← **Less Obvious**

| Communities

- **Traditional Engineering Teams**

- IaC, CI/CD, DevOps & SysOps help

← **Most Obvious**

- **Data Analysts & Scientists**

- MLOps, Specialised HW/SW (GPUs etc),
Version Tracking (Code,Data,Parameters)

← **Less Obvious**

- **Leadership & Governance**

← **Sometimes Forgotten,
Sometimes Over Emphasized**



| Communities

- **Traditional Engineering Teams**

- IaC, CI/CD, DevOps & SysOps help

← **Most Obvious**

- **Data Analysts & Scientists**

- MLOps, Specialised HW/SW (GPUs etc),
Version Tracking (Code,Data,Parameters)

← **Less Obvious**

- **Leadership & Governance**

- FinOps, SecOps etc

← **Sometimes Forgotten,
Sometimes Over Emphasized**



| Communities

Recognise & take all your communities into account

Feature & Product Teams may well be your obvious primary users, but they're not the only ones



3

Characteristics Of A Successful Platform Experience



| Successful Platform Experience Characteristics



Clear Boundaries & Responsibilities

- Minimise Frustration For Engineers
- Promotes Better Collaboration & Flow

| Successful Platform Experience Characteristics



Clear Boundaries & Responsibilities

- Minimise Frustration For Engineers
- Promotes Better Collaboration & Flow

Self Service & Automation

- Empowers Independent Teams
- Promotes Innovation & experimentation

Successful Platform Experience Characteristics



Clear Boundaries & Responsibilities

- Minimise Frustration For Engineers
- Promotes Better Collaboration & Flow

Self Service & Automation

- Empowers Independent Teams
- Promotes Innovation & experimentation

Flexible & Evolvable

- Not A One-Size-Fits-All
- Fit For Purpose Tech & Processes

Successful Platform Experience Characteristics



Clear Boundaries & Responsibilities

- Minimise Frustration For Engineers
- Promotes Better Collaboration & Flow

Self Service & Automation

- Empowers Independent Teams
- Promotes Innovation & experimentation

Flexible & Evolvable

- Not A One-Size-Fits-All
- Fit For Purpose Tech & Processes

Reliable & Caters for Day 2 Operations

- Confidence for Engineers
- Confidence for Management



5

So How Do You Platform Engineer Well?



| Prerequisite: Platform Success Requires

**Exec / Leadership Buy-In AND
Technical Experience & Expertise**

- Ensuring both tech & non tech changes can be implemented effectively



| Adjust Your Thinking:

Aim Not Just To *Provide* A Service
But Rather To *Be* A Service
(& Serve Your Communities)



| Community Driven Principles

- Make Teams *Independent of* Not Dependant on you
- Promote *Freedom* Not Autonomy
- Be A *Role Model* & Walk The Talk
- Respect & Recognise Community Differences



Make Teams Independent Of You



| Make Teams Independent Of You

Self
Service

Clear
Bound...

Flexible &
Evolvable

Reliable

- **Define A Platform Contract**
(Like AWS Shared Responsibility Model)
 - Security & Compliance
 - Infrastructure Provisioning



| Make Teams Independent Of You

	Self Service	Clear Bound...	Flexible & Evolvable	Reliable
■ Define A Platform Contract (Like AWS Shared Responsibility Model) ○ Security & Compliance ○ Infrastructure Provisioning		✗		
■ Favour Automation & API Interactions ○ Onboarding, Infra Provisioning ○ Terraform Modules, K8S Operators	✗	✗		✗

| Make Teams Independent Of You

	Self Service	Clear Bound...	Flexible & Evolvable	Reliable
■ Define A Platform Contract (Like AWS Shared Responsibility Model) ○ Security & Compliance ○ Infrastructure Provisioning		✗		
■ Favour Automation & API Interactions ○ Onboarding, Infra Provisioning ○ Terraform Modules, K8S Operators	✗	✗		✗
■ Documentation ○ Includes In-Tool Help & Reference Impls	✗			



Promote Freedom Over Autonomy



| Promote Freedom Over Autonomy

Self
Service

Clear
Bound...

Flexible &
Evolvable

Reliable

- **Establish the Ground Rules**
 - Single Cloud vs Multi Cloud
 - Support for “stable” vs “latest”



| Promote Freedom Over Autonomy

	Self Service	Clear Bound...	Flexible & Evolvable	Reliable
■ Establish the Ground Rules		✗		
○ Single Cloud vs Multi Cloud				
○ Support for "stable" vs "latest"				
■ Choice Over "Anything Goes"		✗	✗	✗
○ Tech Stacks				
○ Ecosystems (Terraform, K8S)				
○ Templated Pipelines				

Be A Role Model, Walk The Talk



| Be A Role Model, Walk The Talk

Self
Service

Clear
Bound...

Flexible &
Evolvable

Reliable

- **Eat Your Own Dogfood**
 - Tooling (using your own stuff)
 - Test against Reference Implementations



| Be A Role Model, Walk The Talk

Self
Service

Clear
Bound...

Flexible &
Evolvable

Reliable

- **Eat Your Own Dogfood**
 - Tooling (using your own stuff)
 - Test against Reference Implementations
- **Offer “Professional Services”**
 - Workshops, pairing, embed in team



| Be A Role Model, Walk The Talk

	Self Service	Clear Bound...	Flexible & Evolvable	Reliable
■ Eat Your Own Dogfood ○ Tooling (using your own stuff) ○ Test against Reference Implementations				✗
■ Offer “Professional Services” ○ Workshops, pairing, embed in team			✗	✗
■ Platform Evangelists / Advocates ○ Promote Good Practices		✗		✗

Respect & Recognise Community Differences



| Respect & Recognise Community Differences

Self
Service

Clear
Bound...

Flexible &
Evolvable

Reliable

- **Adapt Ways Of Working For Different Communities**
 - Team Structures & Interaction Modes



Respect & Recognise Community Differences

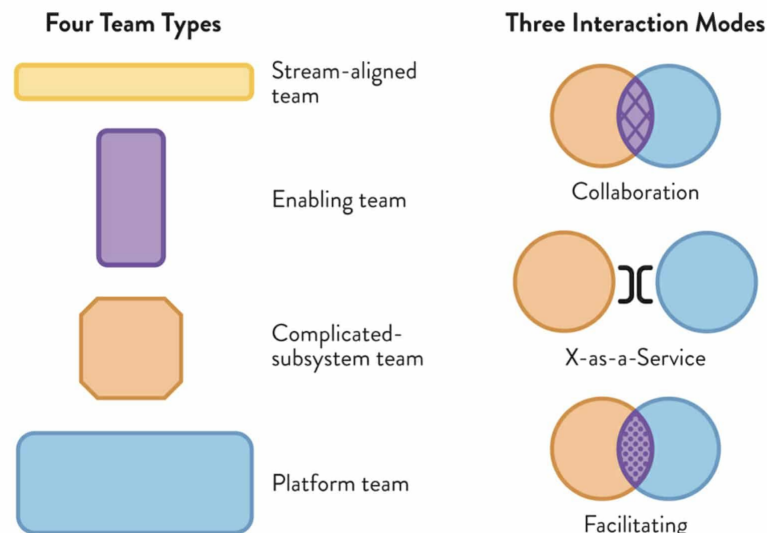
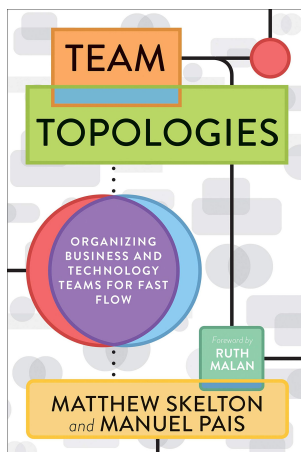
Self
Service

Clear
Bound...

Flexible &
Evolvable

Reliable

- Adapt Ways Of Working For Different Communities
 - Team Structures & Interaction Modes



| Recap: Community Driven Principles

- **Make Teams *Independent of* (Not Dependant on) you**
 - Define A Platform Contract
 - Favour Automation & API Interactions
 - Provide Decent Documentation
- **Promote *Freedom* Not Autonomy**
 - Establish The Ground Rules
 - Choice Over “Anything Goes”
- **Be A *Role Model* & Walk The Talk**
 - Eat Your Own Dogfood
 - Offer “Professional Services”
 - Consider A Platform Evangelist
- **Respect & Recognise Community Differences**
 - Adapt Ways Of Working For Different Communities & Teams



Conclusion

- **Start With A Good Foundation**
(Exec Buy-In & Technical Expertise)



Conclusion

- **Start With A Good Foundation**
(Exec Buy-In & Technical Expertise)
- **Guided By Community Driven Principles**
 - Make Teams Independent Of You
 - Promote Freedom Not Autonomy
 - Be A Role Model & Walk The Talk
 - Respect & Recognise Community Differences



Conclusion

- **Start With A Good Foundation**
(Exec Buy-In & Technical Expertise)
- **Guided By Community Driven Principles**
 - Make Teams Independent Of You
 - Promote Freedom Not Autonomy
 - Be A Role Model & Walk The Talk
 - Respect & Recognise Community Differences
- **Aim To Create A Platform With**
 - Clear Boundaries & Responsibilities
 - Self Service & Automation
 - Flexible & Evolvable
 - Reliable



Conclusion

- **Start With A Good Foundation**
(Exec Buy-In & Technical Expertise)
- **Guided By Community Driven Principles**
 - Make Teams Independent Of You
 - Promote Freedom Not Autonomy
 - Be A Role Model & Walk The Talk
 - Respect & Recognise Community Differences
- **Aim To Create A Platform With**
 - Clear Boundaries & Responsibilities
 - Self Service & Automation
 - Flexible & Evolvable
 - Reliable



Questions?

OpenCredo

A TRIFORK COMPANY



info@opencredo.com



www.opencredo.com

