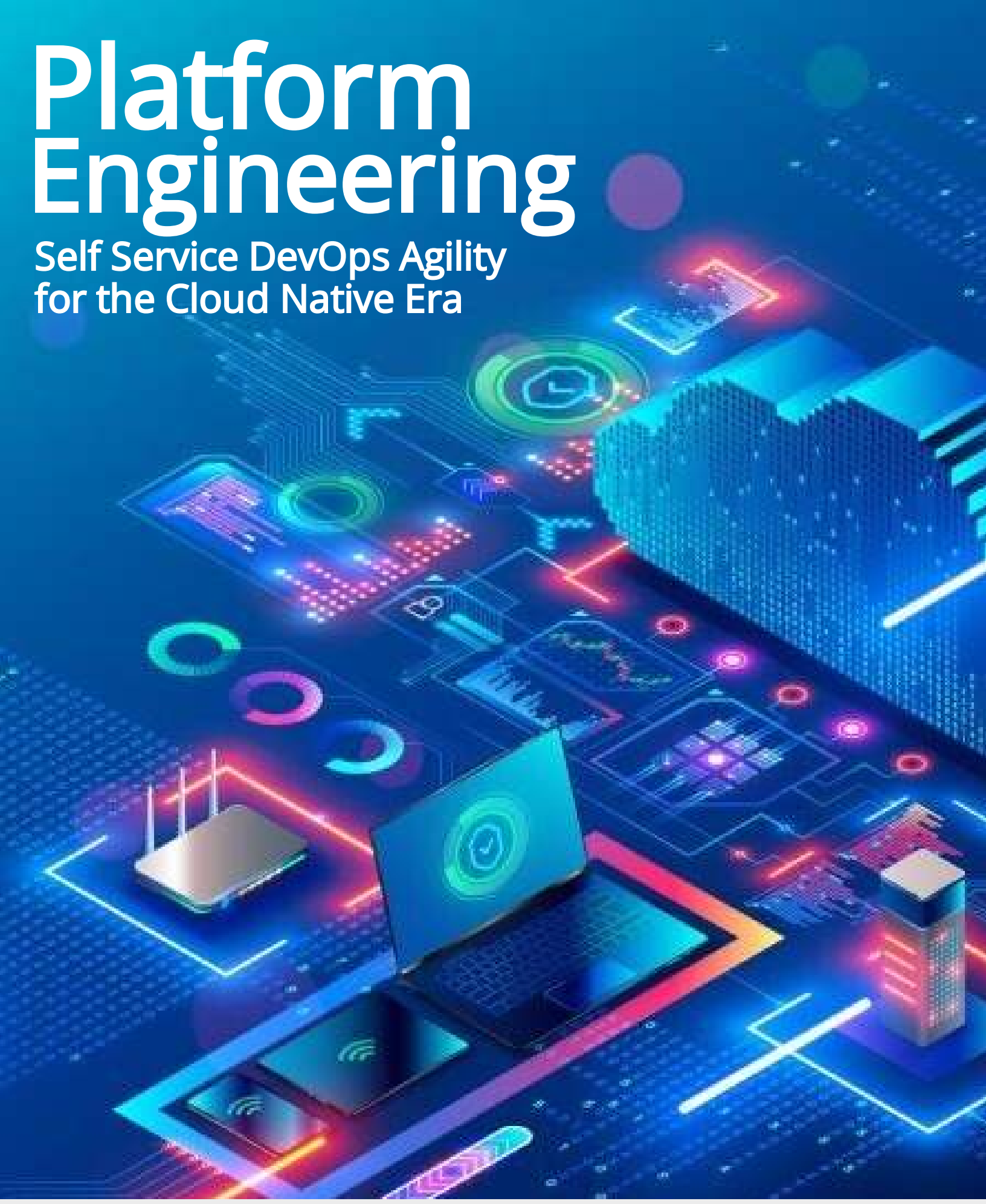
# Platform Engineering

## Self Service DevOps Agility for the Cloud Native Era



DevOpsFlow.net

# What is 'Platform Engineering'? - DevopsFlow.net

## As specialist web site PlatformEngineering.org defines:

“Platform engineering is the discipline of designing and building toolchains and workflows that enable self-service capabilities for software engineering organizations in the cloud-native era.”

Gartner expects that by 2026, 80% of software engineering organizations will establish platform teams as internal providers of reusable services, components and tools for application delivery.

Notably this is headlined by ‘IDPs’ – Internal Developer Platforms: “An IDP consists of many different techs and tools, glued together in a way that lowers cognitive load on developers without abstracting away context and underlying technologies.”

Platform teams support this by delivering shared services across each layer of cloud infrastructure, helping product teams innovate and iterate at speed.

Our ebook offers a practical blueprint for adopting an Enterprise Platform strategy, through documenting keynote case studies such as the HMRC’s Multi-channel Digital Tax Platform, and implementation guides for vendors like Microsoft Azure.

# HMRC: Hyperscale PaaS for Hyperscale Performance

## Speaking at the DevOps Enterprise Summit HMRC presented on: “Saving the Economy From Ruin with a Hyperscale PaaS”.

The talk was delivered by:

- Ben Conrad – Head of Agile Delivery – HMRC.
- Matt Hyatt – Technical Delivery Manager – Equal Experts.

## Crisis and Agility

In March 2020 the United Kingdom went into pandemic lockdown, causing the most brutal recession in living memory. The Government had to react quickly, with policies to help citizens and businesses cope.

This required HMRC to rapidly implement a range of new services, such as The Coronavirus Job Retention Scheme, The Self Employment Income Support Scheme and the Eat Out to Help Out Scheme.

Once these initiatives were announced, the UK’s Tax Department (HMRC), needed to encode these policies as digital services, capable of dealing with huge spikes in traffic. And it had to be designed, implemented and delivered in a matter of weeks.

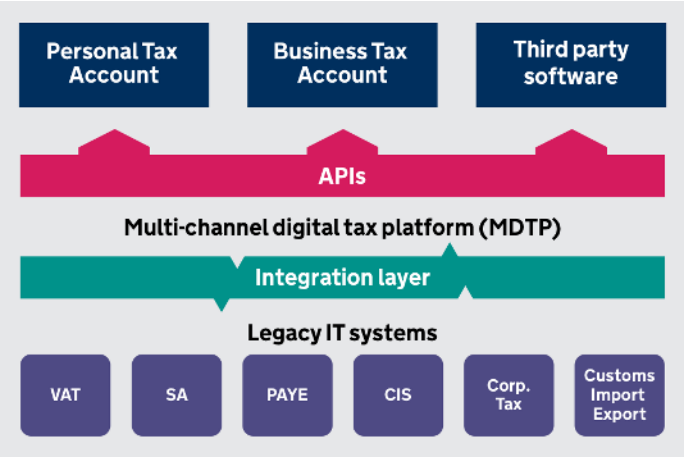
The platform and services needed to deliver financial support to more than 12 million employed and self-employed workers via the three schemes outlined above. In just a matter of weeks, the team was ready and the platform withstood 67,000 job claims within half an hour of the Job Retention Scheme going live. It also handled 440,000 applications for government grants via the Self Employment Income Support Scheme on the first day of its operation.

# HMRC: Hyperscale PaaS for Hyperscale Performance

## Platform Engineering

What made this possible was HMRC's Platform strategy. Their Multi-channel Digital Tax Platform (MDTP) – which employs Continuous Delivery at scale, began development in 2014. It is a cloud platform for over 130 user-facing applications, powered by over 1000 microservices that have been built as part of HMRC's 'making tax digital' strategy. It provides an easy way for teams to build and deploy applications that can scale to handle millions of requests.

This enables business domains within HMRC to fund a small cross-functional team to develop a microservice or collection of microservices, rapidly developing and deploying new customer facing services. The platform provides a suite of common components that makes this possible.




## Delivery at Scale

HMRC is the largest digital platform in the UK Government, and the platform enables the very large scale delivery, encompassing:

- Hosting of 1,200 microservices.
- Built by 2,000+ developers, split into 70 teams across 8 geographic locations.
- Making over 100 deployments per day.

# HMRC: Hyperscale PaaS for Hyperscale Performance

New digital services are developed by a cross-functional team of developers, interaction and content designers, QAs, business analysts et al, and the platform provides all the functionality for the team to create and deploy services such as Github, automated pipelines and telemetry tooling and dashboards. It also includes collaboration apps.

## Guardrails

The central principle to success is the concept of an “opinionated platform”, or simply guardrails. Basically this means a set of rules that must be followed, that are ‘baked in’ to the platform, and by operating within those rules developers are then free to work rapidly and make large volumes of changes without approval. Example rules include:

- Microservices must be written in Scala, using the Play framework.
- Persistence must be achieved by using Mongo.

The primary benefit of this platform approach is the ease by which they could meet the dramatic scale required. Having a platform composed of immutable infrastructure defined as code meant they could size the platform to meet the size of demand.

Given the scale of the traffic due to the pandemic situation there were still key challenges. For example HMRC could take steps to assure and test performance of their own apps, but when they relied on third parties via APIs they could not, so they had to take steps such as setting these up as asynchronous calls.

The other key challenge was MDTP had to interface to HMRC’s legacy application estate, mostly still running on mainframes. This was addressed by migrating data from these systems into temporary data stores on MDTP.

# HMRC: Hyperscale PaaS for Hyperscale Performance

## Conclusion

MDTP offers an exemplar blueprint for an enterprise Platform strategy.

This is a powerful story of how the right combination of culture, technology and focus can empower a large organisation to pivot fast and efficiently, resulting in the rapid delivery of digital services that are user-centric, maintainable, performant and resilient.

# Azure Platform Engineering: Best Practices, Blueprints and Vendor Capabilities

At 10m:50s of his interview Mark Russinovich, Azure CTO, describes how 'Platform Engineering' is emerging as the best practice for empowering more productive developers.

He is explaining Radius, a new application platform they have released to support the development and deployment of Cloud Native applications.

Radius is intended to facilitate the abstraction of applications from their underlying infrastructure so that the applications can be commonly deployed to any Kubernetes environment.

Enterprise developers want to focus on their code writing without also having to learn everything about Kubernetes too, and so Radius frees and empowers developers from this requirement. Hence Radius is an ideal mechanism for building environments intended to enable Platform Engineering practices.

In short it's for the core component of an 'IDP' – Internal Developer Platform, the central feature of Platform Engineering. As PlatformEngineering.org describes:

*"Platform engineering is the discipline of designing and building toolchains and workflows that enable self-service capabilities for software engineering organizations in the cloud-native era."*

In the feature video Dan Sol and Mark Weitzel of Microsoft share their vision for Platform Engineering, describing that in essence it is the matured evolution of DevOps, building on top of it to address the exploding complexity of modern Cloud and developer technologies. They share insights covering:

# Azure Platform Engineering: Best Practices, Blueprints and Vendor Capabilities

- How to establish Platform Engineering as an organizational capability.
- Microsoft's own adoption journey and the technologies and architecture used.
- What Metrics and KPIs can guide the success of an IDP strategy.
- Core design principles for implementing Platform Engineering.
- How to build a pre-configured application hosting environment.

Another Microsoft expert talk explores these practices further. Self-Serve App Infrastructure explains how using Azure Deployment Environments empowers devs to quickly deploy the right environment at the right time using standardized, project-based templates.

In this insightful video, Kaspar Von Grünberg from Humanitec shares practical guidance and best practices for architecting an enterprise-grade IDP, explaining the core design principles including golden paths, standardization, dynamic configuration management, developer workflow freedom, and code as the single source of truth.

## Azure Reference Model

Furthermore Kaspar offers this reference model for a specific implementation on Microsoft Azure, explained in this video presentation, where he walks through the architectural components of a modern dynamic IDP based on real-world experiences:

# Azure Platform Engineering: Best Practices, Blueprints and Vendor Capabilities

- Learn design principles for a scalable IDP using interchangeable tools such as Azure EKS, Humanitec, Azure Pipelines, Terraform, and more.
- Design great interaction patterns for platform engineers and developers to standardize config files and enable true developer self-service.
- Solve common challenges such as low levels of productivity and innovation, missing self-service, cognitive developer load, and lack of standardization through transforming your static CI/CD setup into an enterprise-grade IDP.

Download the presentation slides here.

## Featured Partner: Northflank

As they **describe here** Northflank provides a DevOps solution ideal for enhancing the self-service developer experience, or delivering an Internal Developer Platform (IDP) or Application Delivery Platform (ADP) on Azure Kubernetes Service (AKS).