cycloid

IN IT FOR THE LONG HAUL

Platform Engineering in the Age of Sustainability

TABLE OF CONTENTS

PROLOGUE

15 years after the arrival of DevOps and things still aren't working out – end-users and developers are drowning in a sea of tools, automation and clouds. The Ops team is buried under an avalanche of tickets - and management? We've heard it can be a bit like steering a ship while blindfolded and with your hands tied behind your back! Meanwhile, irresponsible cloud use drains IT budgets faster than you can say "sustainability", resulting in wasted resources and an enormous cloud carbon footprint.

Since Gartner validated the term "platform engineering" as a practical approach to improving the developer experience and productivity by automating infrastructure management, Cycloid's ambition has been twofold: to take leadership in the platform engineering category and make platform engineering sustainable. Why? Because by putting sustainability on the orchestration level (when it's at the core of all major IT processes), you can make a bigger, better difference to your environmental impact.

"Sustainable platform engineering" won't turn your business into a zero-emissions enterprise overnight. The IT sector is huge and there are too many variables. However, you'll see that decreasing – or at least controlling – cloud infrastructure usage is a solid environmental topic that individual companies can tackle by making changes to infra management

This is where platform engineering comes into play. Implementing an internal developer platform that acts as a cornerstone to all your applications, tools, and cloud will also allow you to keep fine-grained control over deployments and users, as well as which cloud will be used and how. This will bring order to your processes and make happier devs, and will also help reduce cloud consumption and eliminate waste.

In this ebook, we'll be going over some ways to improve your infrastructure management in preparation for the creation of a platform, create a better self-service experience for end-users, and implement governance guardrails to keep your resource and cloud usage in check. Sustainable platform engineering will encourage a culture of sobriety that flows through your organization from the bottom up, enabling smarter, more environmentally-conscious cloud consumption at every level and, even better, more efficient processes for your teams.

CHAPTER 1

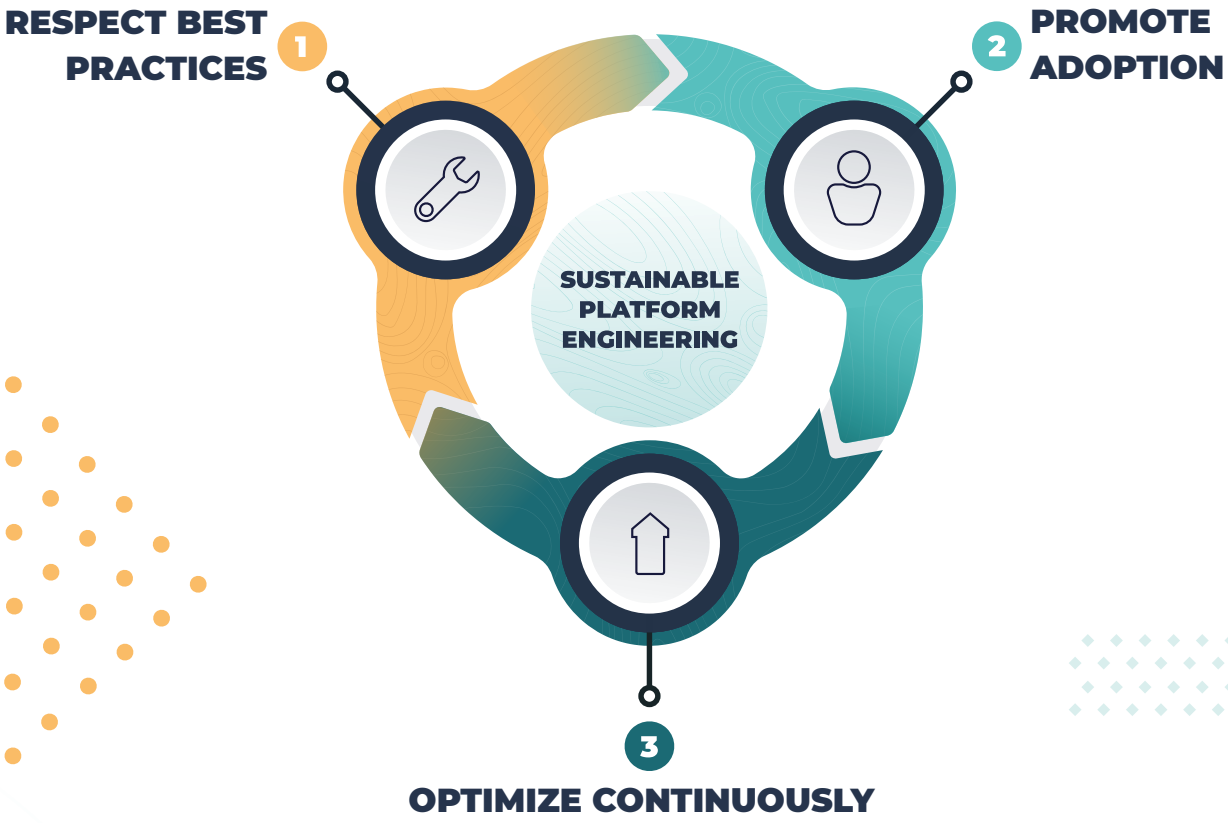
What is sustainable Platform Engineering?

So, let's start with the basics. To explain this one, we're going to bring sustainability back to its roots. In general, sustainability means meeting the needs of the present without compromising the ability of future generations to meet their needs.

The definition comprises three pillars: social, environmental, and economic — also known informally as people, planet, and profit.

In the context of platform engineering, this means building a foundation and tools that are going to last and deliver a stable return on investment year after year.

Okay, but practically, how do I achieve sustainability?



When it comes to your platform tool, it isn't only the making of the beast that's the killer - even if you have a strong vision with the right people and team in place. It's the maintenance, upkeep, and usability of the platform as your business moves forward. This takes time, effort, and money but, in today's business environment, you can't just throw these variables at the problem and hope to see radical results. Instead, you need to act strategically, choosing paths to your platform that will sustain your business in the long run.

Focus on:

- **GitOps/Infra as code** - Optimize your infra, modernize it, and enjoy the benefits. Together, IAC and GitOps are what allow you to pick up speed and simplicity while maintaining a flawless end product.

- **Self-service** - A better self-service tool with great UX empowers users and takes the pressure off the platform team, SREs, DevOps and cloud experts, leaving both with more headspace and free time to concentrate on bigger questions.

Governance guardrails - Your portal will bring flexibility to end-users while decreasing the number of tickets for the platform team. Do it right and it will also keep governance in check and control your cloud spending and carbon footprint impact, offsetting the environmental impact of your business while continuing to optimize people and profits.

Why Platform Engineering Needs GreenOps

It's really easy to conflate GreenOps and sustainability, and they're certainly closely related. That's why, before we get into the specifics, we need to make sure that everyone's clear on exactly how they differ.

Sustainability is much wider-ranging than GreenOps, while GreenOps is an operating model - albeit an important one - in the sustainability arsenal. For many companies, sustainability still isn't a major concern, whereas, in others, greenwashing is being used to cover a multitude of sins.

In the world of IT, this is unacceptable - IT-based carbon emissions from cloud computing have overtaken emissions from big baddie commercial aviation. Unsurprisingly, most emissions come from inefficient cloud use due to overly complex processes and a lack of centralized oversight over cloud infrastructures.

And if "emissions" sounds vague to you, here's a translation – a whopping $147 billion was wasted in 2022 in cloud spend.

Putting things in financial terms does seem to be a failsafe way to get people to pay attention. That's where the concept of FinOps comes into play, obviously - if the environmental impact isn't making the mark it should on the powers that be, then the concept of throwing away money in the form of wasteful configuration and use of cloud resources should do the job. Either way, the earth's reaching a crisis point and we firmly believe that by encouraging sustainability across the whole enterprise, you'll hit both birds with one stone.

Oh, and you need to do it now.

Taking charge of your cloud carbon footprint may seem like a daunting task, but in reality it's easier than you think! Here are 6 practical ways you can reduce your cloud usage (and your cloud bills!)

Download the infographic

Next, we're going to look at some practical ways to actually improve platform engineering and act as a lynchpin in the search for a better, more sustainable approach to cloud computing.

CHAPTER 2

Improve infrastructure management in preparation for the creation of a platform

Modernizing the way you deal with your infra, at least at the beginning of the process, isn't exactly hi-tech. Even so, we think it's one of the best first steps you can take - think about building bases and establishing sustainable foundations for tomorrow's business, not building sky-high and adding bells and whistles today. Modernizing is about infrastructure-as-code (IAC) and using that IAC to automate as much as possible, streamlining processes and eliminating human error. This is something that works especially well in a GitOps context, which we'll also talk about.

The foundation: Infra-as-code

We're unlikely to have to convince you of the benefits of infrastructure as code, but if we're still having this conversation, we might have to convince you of the benefits of biting the bullet. Sure, there are blockers - it's unlikely to provide full coverage of your infra, and legacy infra could cause a problem, but making sure you've got the greatest coverage possible leaves much more time and space for higher-level tasks. Let's keep out of the description configuration management, CI/CD pipelines, HELM etc, it is the same approach that apply.

The system: GitOps

Once infra-as-code is underway and, CM, CI/CD, etc are ongoing, you'll be in a much better position to make the move to GitOps. It's the natural extension of the idea underpinning all of our modernization - use the available tech to remove manual involvement and human error.

By making your Git the single source of all truth and home to your code, config, and infrastructure (in the form of IAC), you can set up the pipelines you need to deploy and manage your infra and applications. It further automates the day-to-day of your environments, as well as ensures that there's always a single source of truth (the infra in your Git) to refer back to, and helps with security and other requirements for your code by checking and applying requirements as part of the continuous deployment process.

By using Git, you can take advantage of versioning and rollback capabilities, visibility and observability, and the ability to apply security requirements in a timely manner.

Also:

- **Creates a standard workflow**
- **Enables consistency across environments**
- **Leaves existing code repos intact**

The future: the platform engineering team

Once these two linked tasks are completed (a move to IAC and then implementing the use of Git to manage it), you've got a much more manageable technology "package" to entrust to the platform team.

Even so, always remember that…

GitOps alone isn't enough

GitOps is a really great starting point for sustainability - it offers observability, sustainability, and rock-solid governance, meaning that under your watch, cloud (and therefore carbon) wastage will be easy to keep under control. There's a but...GitHub is a tech tool for tech people - but sustainability is something that affects the whole team. That's why we qualify it as an essential starting point, but something that you'll need to add to make it truly a tool for everyone on the team.

Whether you join the dots yourself or use a 3rd party tool, you'll need to make sure that any part of your platform that's going to be accessed by non-technical people has an interface that you don't need AWS (or GCP, Azure, VMWare, Terraform, Ansible, or Kubernetes) certification to get your head around.

Once it's up and running, you'll see that GitOps is a step in the right direction. But you'll also realize that it's not an end in itself - it's more like a step in the process. Very soon, a self-service portal will begin to look very much like the most sensible evolution of the journey.

CHAPTER 3

Create a better self-service experience for end-users

Building a platform engineering portal is a colossal waste of money and time - and therefore very unsustainable - if no one uses it. You need to make sure the possible dealbreakers are addressed long before you ever make a move or else sustainability will be nothing but a pipe dream. After all, building your own portal is a massively time-consuming process, so wouldn't it be great if you could guarantee a successful platform before you went to the effort?

Don't believe us? Just take a look:

That's why, on average, it takes a whopping 3 years for platform engineering to be able to show a positive impact on the business.

Bear this in mind...

It's really important that you maximize the chances of your developers wanting to use the portal and even more so before you start to build it. If we had to boil this down to one piece of advice for how to make this likely, it would be this: the dev is your end customer. Max out the DevEx and the users (and evolution) will follow.

Max out the DevEx

Focus on making your IDP the best environment for your end customers (the development team) to do great work and have a great time doing it. To do this, especially if platform engineering is a new concept in your organization, you need to make sure that the people around and above you understand that the devs are your end customers, not the product.

Make it truly self-service

We've found this to be one of the most important parts of the platform - you need to make it truly self-service. If you don't, you'll miss a huge proportion of the objective - and the ability to hook your audience. The problem here is that although most people acknowledge that self-service is a great thing, it's actually pretty hard to provide through entirely internal development - or at least, it requires a lot more work. By outsourcing some of the development processes to a 3rd party tool, you can focus on the specifics of your business, while entrusting the bells and whistles to people whose only job is to make your people's lives easier.

Build for purpose

This one is a no-brainer, but when there are a lot of spoons in the pot, it can get lost in a mire of opinions, inheritances, and ill-advised professional interests. How so? The idea of an IDP probably isn't new. There may be traces of previous plans, tools that were trialled, and ideas that were considered and discarded, some or all of which may resurface during the planning of the new platform. Don't let strong opinions or easy options knock you off course - your IDP needs to be perfectly aligned with the needs of your current team, for the challenges of today (and tomorrow).

Employ professionals of usability

If you were building a product for an external audience, you wouldn't even consider unleashing a tool that had no usability auditing built in - so why would you unleash such a beast on your team? Make sure UX plays a central role in the development process or ensure that your 3rd party tool has a rock-solid UX foundation.

Lessen the donkey work and remove complexity

We've already mentioned automation and how it comes later in the process to win fans and admiration, but that won't happen spontaneously. Make sure the intention to lessen repetitive, boring, but necessary work is a central part of your platform. Taking full advantage of IAC, CM, CI/CD, Helm, etc will ensure you're in a great position to use ready-built tools to take on some of this burden.

CHAPTER 4

Implement governance guardrails to keep your resource and cloud usage in check

One of the most important ways to defuse possible pitfalls is the one we'll discuss right now - governance and security. In this domain, balance is everything and if your team have so much of an inkling that it's off-kilter, your platform will be sunk before you can say "principle of least privilege". The overall aim for security and governance in platform engineering is to keep things tight enough to keep ops happy and cloud consumption in check, but flexible enough to keep users satisfied at the same time.

In many organizations, however, governance is still a bit of a dumpster fire. There's a lack of clarity and a sense of randomness about policies and best practices and, as a result, infra management and resources. Even so, it's also one of the few places where you can have a clear and direct impact on sustainability - clear up the mess, impose order, and start proactively streamlining and optimizing your usage. But when it comes to imposing order, there's a risk of the extremes we spoke about earlier - and it's that risk that might undermine the usability of your platform.

Getting the balance right

There's a sweet spot, a compromise between the needs of Ops and the experience you offer your users - get it right, and your portal will thrive. Here's what we think you need to concentrate on:

Aim for a center of excellence

Don't just aim for organization, aim for a center of excellence where your organization's best practices - like landing zones, architecture, and security - are defined. Once established, it will be easier to teach these policies and procedures to everyone on the team. Remember, true sustainability is a constant game of optimization and iteration - if your team have a clear vision of "excellence", it will mean that no matter the modification, there's a clear method guiding and supporting you.

Ensure a clear hierarchy

Before you actually start tweaking policies, you'll need to establish a clear hierarchy of permissions and privileges per user, team, organization, or account. You can establish these permissions in whatever format you like - the main thing is that you execute the hierarchy and stick to it in the future. It's really important that this is carried out before you actually start configuring, to make sure that everything is crystal clear and strategically planned before you build anything into the configuration.

Enable 360° visibility

Once you start to build (or use) your new platform, it's essential that you enable some functionality that allows 360° visibility on projects, asset inventory, automation, CI/CD pipelines, events, infrastructure diagram, documentation, and cost and carbon footprint management, or as close to it as possible. It's only through this functionality that you can truly give your team the best chance at maintaining long-term optimization. What you can't see, you can't measure, and nowhere is this more obvious than on your infra.

Perfect quotas, automatic approvals, and role-based permissions

When you have well-controlled governance, it becomes easy to set up appropriate quotas, approvals and permissions, even when you have multiple roles or even businesses to deal with. This removes risk and decision fatigue from non-expert members of your team. On a larger scale, they also contribute to sustainability, since you can keep precise and proactive track of resources, which in turn avoids waste and helps you optimize spending.

How does Platform Engineering help you optimize?

Gartner described platform engineering as a strategy aimed at "improving developer experience and productivity by providing self-service capabilities with automated infrastructure operations." Break platform engineering down, and you'll see how it optimizes in not one but three separate ways.

1. Improved developer experience creates happier, more productive devs. In fact, James Governor expanded on his definition of DevX as "creating an environment where developers can do their best work. Their best work results in optimized productivity and no businessperson needs an explanation as to why that's a good thing.
Sustainability of people

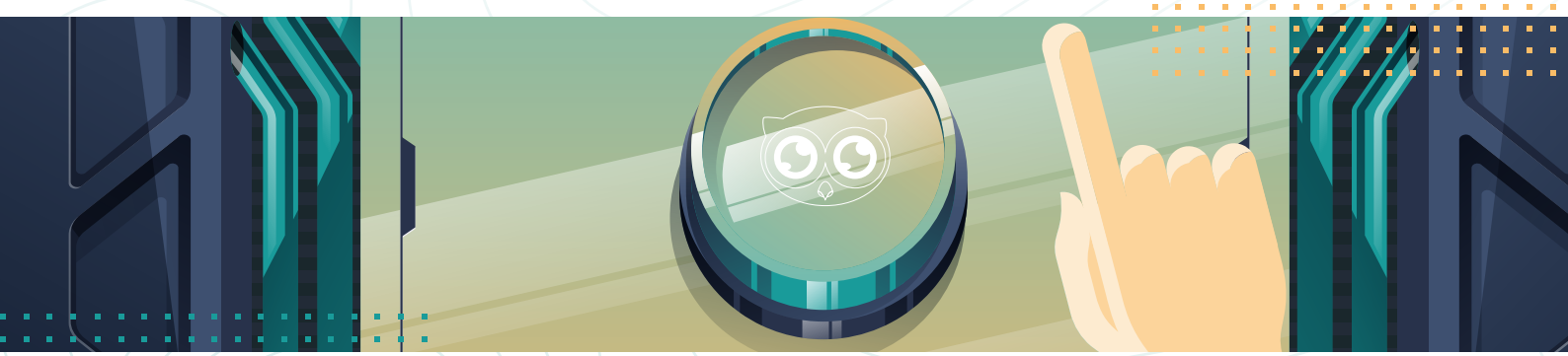
2. Providing self-service capabilities gives you an excellent opportunity to further optimize your developer experience (in fact, can it happen without self-service? We're not sure.) but also gives you an excellent platform to apply safeguards and guide rails that protect both inexperienced users and your cloud consumption (as well as across the board security and governance).
Sustainability of both profits and planet

3. Automated infrastructure operations further rationalize cloud consumption, carbon footprint, and costs as your ops or platform team enables the quickest, most efficient way to carry out everyday essential tasks. No more user error, no more accidental inefficiencies.
Sustainability of both profits and planet

Break down the optimizations, and then you'll see how they are also the quickest way to sustainability, of the **profits**, **planet**, and **people trifecta**.

Ergo, platform engineering is the quickest way to sustainability.

What are you waiting for?



cycloid

www.cycloid.io