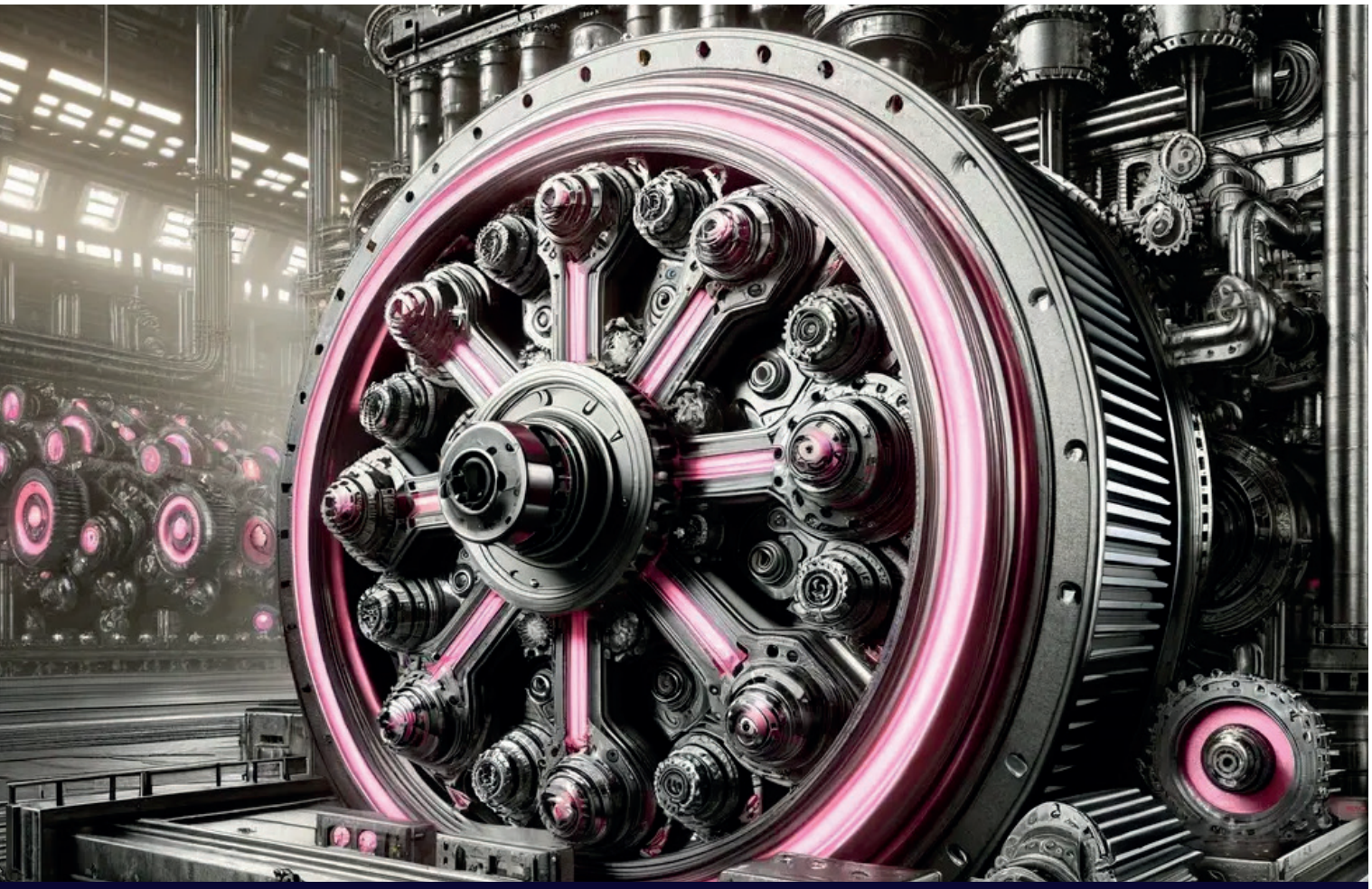
hoopy



A flywheel for developer relations

SPONSORED BY

Common Room

hoopy

Table of contents

Acknowledgements

Thank you to everyone who shared their insights and helped to review this report, including

- The team at Common Room
- Rebecca Marshburn
- Phil Leggetter
- Jen Sable Lopez
- Carmen Huidobro
- Kevin Lewis
- Sarah Dorward

Design and layout by Kat Wright

SPONSORED BY

Common Room

Introduction

Get enough DevRel people together in the same place and, eventually, the conversation will turn to how we measure what we do. Perhaps that's because, even now, developer relations can mean somewhat different things from one company to the next.

But it might also be that we lack tools for thinking about what we do. Our friends in marketing have many competing frameworks, new books every month, and even doctoral theses to draw on.

Here, I propose another way for us to think about how we plan, measure, and communicate developer relations. It's not an entirely original idea. Flywheels are not new, even in DevRel. But by speaking to developer relations leaders and drawing on my own consulting work, I've put together a suggestion for a DevRel flywheel that might provide a point of reference for DevRel teams considering how better to frame their work.

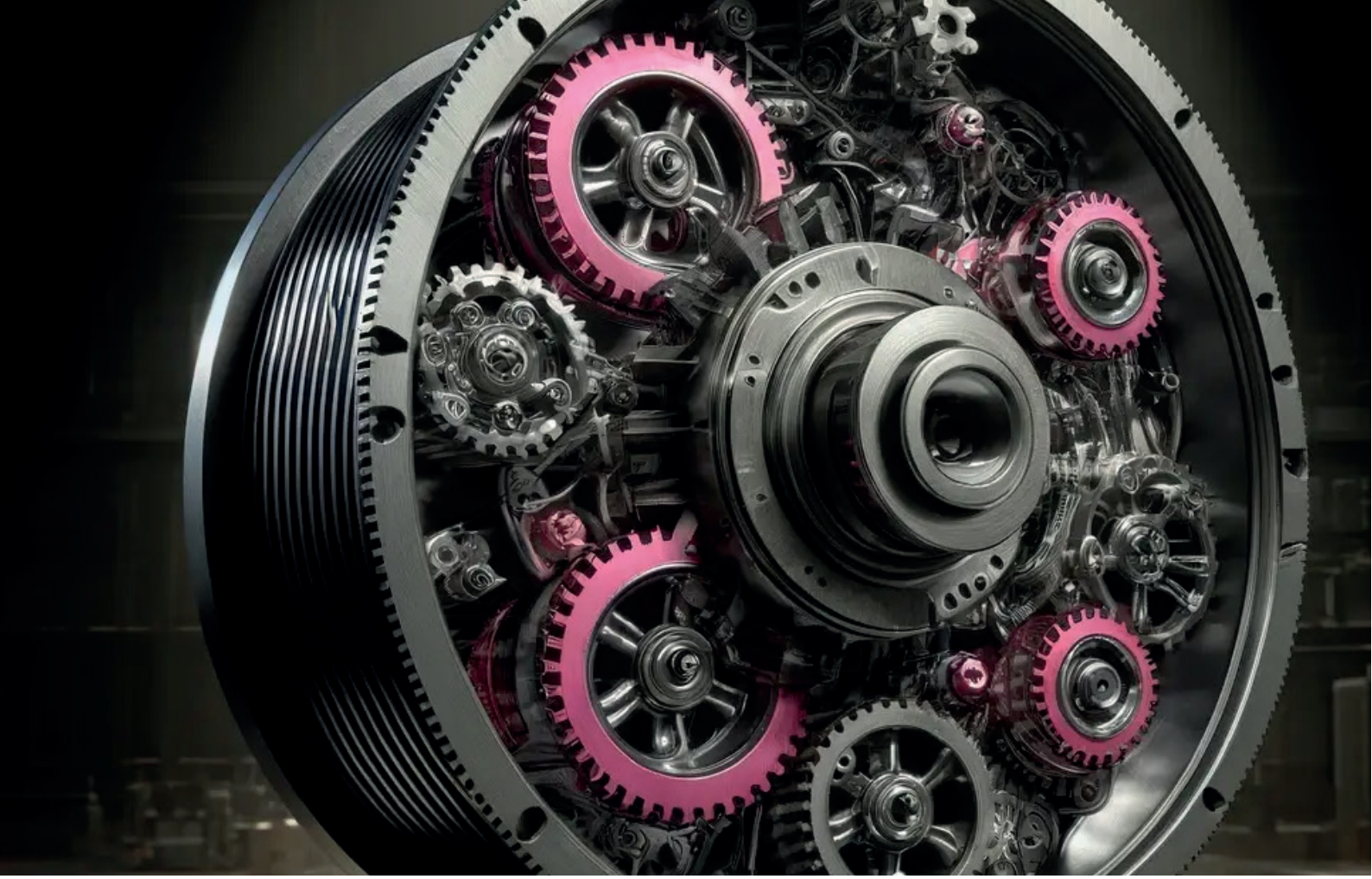
Matthew Revell
Hoopy Limited, June 2024

Common Room

We believe in the power of developer relations—to foster growth, fuel innovation, and drive better business outcomes. It's why we think the discipline deserves attention, insights, and clarity around the role, its responsibilities, and its implementation.

We're proud to support work that contributes to the DevRel knowledge base, and we hope this resource helps you and your organization build and grow your DevRel program with confidence. If you'd like to learn more about Common Room, come see what we do for DevRel teams and say hi any time.

Common Room
June 2024



Frameworks for DevRel strategy

In DevRel, we're lucky. We can borrow from other disciplines, such as marketing and product management. That means we get to use their frameworks and models to plan and measure what we do, alongside a handful that are unique to developer relations.

While a good starting point, these frameworks don't quite capture the self-sustaining cycle at the heart of DevRel. Whether we plan it that way or not, the initiatives we run within our developer relations programs are interconnected. When everythinglines up, each step should amplify the next.

That's why we need a flywheel to help us align our developer relations activity into an engine that multiplies the effort we put in.

There are multiple types of DevRel

There's a problem, though. One of the things that makes DevRel hard is that each program varies so much. Think of some of the reasons that companies invest in developer relations:

- **Drive awareness and adoption**
- **Nurture a contributor community**
- **Train and enable developers**
- **Develop the ecosystem around a product**

And there are more. But for most DevRel programs, that first category of getting people to use a product or technology is either a primary or secondary goal. So that's why the flywheel we're proposing focuses on awareness and adoption.

What is a flywheel?

Imagine the type of steam engine that powered factories and mills at the start of the industrial age. Just like a steam locomotive, those engines produced power in bursts. But bursts of power aren't good for running machinery.

That's where the flywheel comes in. Its job is to smooth out peaks and troughs in power. What makes it a useful device for thinking about DevRel is that flywheels are not just about maintaining consistency; they're about building momentum.

It's hard to get the wheel moving but, in time, the same effort makes it spin faster and faster. Eventually, you can cut the power and the flywheel keeps spinning under its own momentum.

Here's how Jen Sable Lopez, who leads DevRel at Contentful, describes it:

A flywheel stores kinetic energy as force applied, which in turn generates momentum. By eliminating friction along the way, it becomes energy efficient and spins even faster. Add continuous positive energy to the flywheel, and you'll accelerate growth.

By focusing on acquisition, retention, and adding the additional force of advocacy, we create the momentum we need to achieve our growth goals.

As a metaphor, flywheels help us focus on building systems that act as force multipliers. By structuring a program's outputs to feed back into its inputs, a DevRel program should be able to achieve more over time even if your resources and budget stay roughly the same.

Components of a flywheel

Flywheels tend to have four parts:

01 Stages:

These represent the sequential phases developers experience as they interact with your product and community. From initial awareness (Discover) to active advocacy (Advocate), each stage marks a developer's journey toward deeper engagement. Recognizing these stages allows us to tailor our strategies to meet developers where they are, ensuring that we provide the right mix of information, support, and inspiration to move them to the next level of engagement. Common Room helps DevRel teams identify, engage, and measure customers and prospects across the stages of their user journey.

02 Inputs:

Inputs are the energy we invest into the flywheel. They include every resource, activity, and strategy deployed to engage developers at each stage of their journey. This could range from educational content and developer tools to community events and one-on-one support. The effectiveness of these inputs directly influences the flywheel's momentum, meaning we must choose and optimize them carefully to make sure they resonate with the developer's current needs, objections, and objectives.

03 Friction:

In any system, friction represents obstacles that slow down progress. In the context of DevRel, friction can arise from various sources, such as unclear documentation, lack of community engagement, or technical barriers to product adoption. By minimizing friction, we can build a smoother journey for developers as they progress from one stage to the next. Friction isn't wholly negative. It's essential in helping us identify the gaps and mismatches between our strategy and the real world.

04 Outputs:

These are the measurable outcomes of our DevRel efforts. They include increased product awareness and adoption, a vibrant and engaged developer community, and a rise in developers championing our offering. These outputs signify our initiatives' success and feedback into the flywheel as new inputs. Consequently, they reinforce the cycle of engagement and growth and contribute back to business goals and outcomes.

Each component plays a crucial role in keeping our DevRel momentum going. By understanding what each stage entails, we can effectively tailor our developer relations efforts. So, let's break them down.

Other ways of thinking about DevRel

We already have tools to help us think about DevRel strategy and impact. Each tool can help us in different ways, depending on the problem we're trying to solve or the audience we want to communicate with.

So, what are the gaps we're looking to fill?

The Funnel

What it does well:

Aligns with the wider business

Models an individual's journey with our technology and business

Helps model interventions based on what has come before

Where it leaves a gap:

Too linear for most developer relations interactions

Focuses on one goal, such as conversion, at the expense of others

Lacks a plan for people who don't match that goal

AAARRRP

What it does well:

Reflects the different stages of developer adoption

Focuses on driving measurable outcomes

Where it leaves a gap:

Struggles to model the non-linear interactions between developer and product

Orbit model

What it does well:

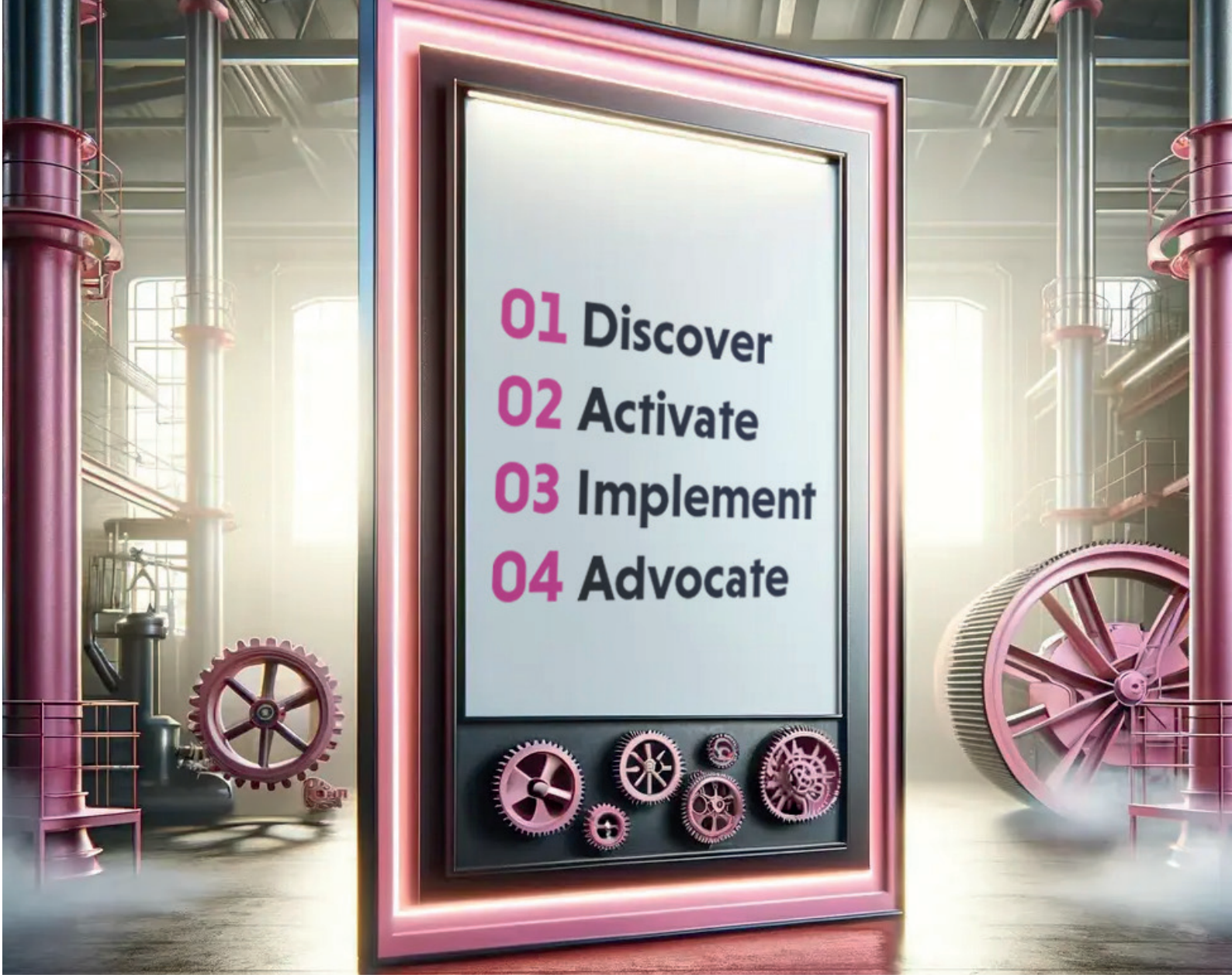
Designed specifically to model community engagement, rather than sales/marketing progress

Useful visualizations

Where it leaves a gap:

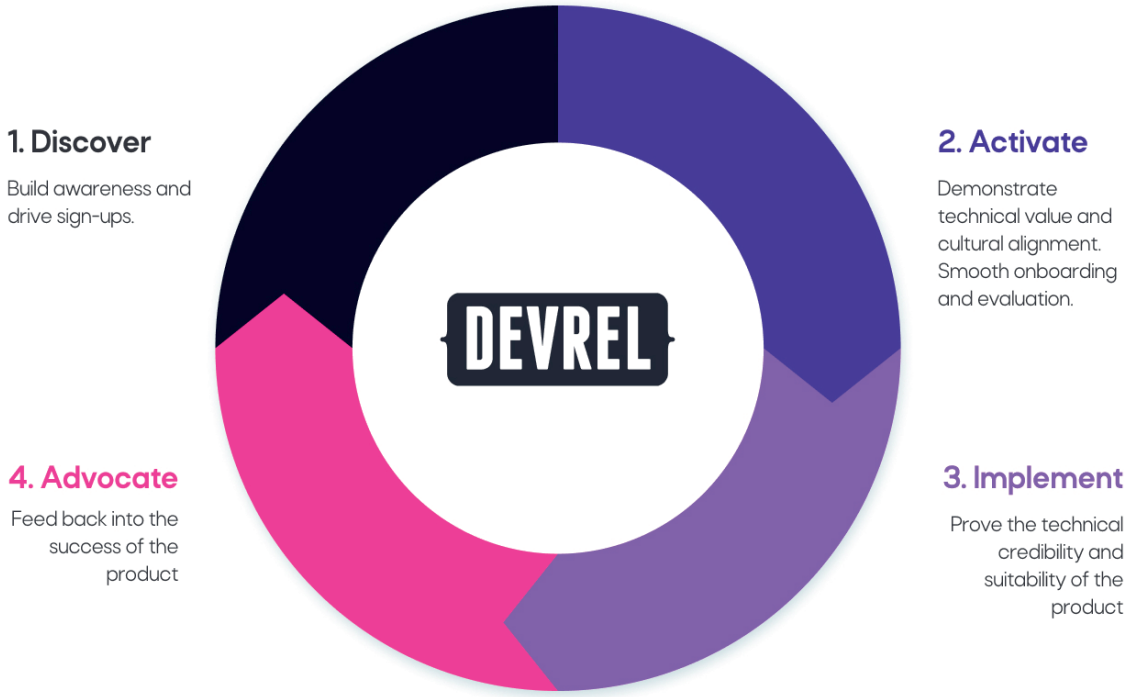
Takes additional work to align it with business outcomes

Not widely understood outside community management circles



Stages of the DevRel flywheel

As a developer becomes more involved with our product, their context and priorities change, too. In response, the initiatives we run need to change, too. Dividing the flywheel into stages helps us to think about what the developer needs and what we should do to drive momentum toward the next stage.



01 **Discover:**

In the **Discover** stage, we reach out to developers through events, content, influencer interactions, organic discussions, and other ways to drive awareness and interest.

02 **Activate:**

Activate is all about building on a developer's initial interest and moving them into evaluation. We can do that by demonstrating the technical suitability of the product and by showing that our company is culturally aligned with developer needs.

03 **Implement:**

The **Implement** stage focuses on all the aspects that take a developer from evaluation through to production. That includes developer experience, community, and support.

04 **Advocate:**

Developers trust developers. In the **Advocate** stage, engaged developers contribute to earlier stages like Discover by offering peer-to-peer recommendations, producing content, and making direct contributions to the product.

Done right, each stage feeds into the next, giving us a self-propelling DevRel program.

DevRel flywheel inputs

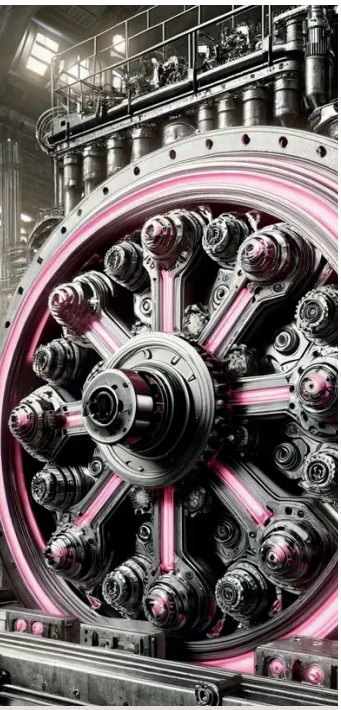
We need to consistently feed in energy to get a flywheel up to speed. That probably isn't a surprise. Neglect one part of your DevRel program for long enough, and you'll soon find that not only does it slow down, but it's even harder to get it up and running again.

Stage	Inputs
Discover 01	• Organic content and SEO • Events, conferences, and meet-ups • SEO/organic content • Paid content placement • Partnerships with other communities • One-on-one outreach • Product marketing content • Influencer activity
Activate 02	• Tutorials, online workshops, and live streams • Sample apps and opportunities to explore • Developer stories and case studies • SEO/organic content • Community activity • Social proof
Implement 03	• Documentation • SDKs and other integrations • Sample apps • Community and ecosystem support • Product and implementation support from the company
Advocate 04	• Champions program • Community nurture programs • Peer-based education programs • Content program • Speaker program • Informal relationship building

DevRel flywheel friction

A physical flywheel has to fight against friction at the point where it meets the axel and also from the air—two easily identifiable points. In DevRel, we're not so lucky. Friction crops up almost everywhere we look.

There are four ways to deal with friction in our DevRel flywheel:



01 Reduce the surface area:

The number of touchpoints in our developer relations program could be a problem in itself. For example, is it hard for your team to manage both a Discord server and a forum? Is that leading to friction from questions going unanswered and community members feeling unappreciated? Focusing your DevRel efforts can reduce the friction that comes from being overcommitted.

02 Minimize drag:

You can only reduce the surface area so far. For those touchpoints that you know are valuable, you can mitigate the friction of engaging with them by using the right tools. Common Room, for example, can improve how you monitor, prioritize, and triage interactions.

03 Apply lubrication:

Some friction comes from poor implementation. For example, let's say you pay community members to write content but your finance team won't pay people outside the US unless they fill out the dreaded W8-BEN-E form. That's likely to impede more casual participants. Find alternative ways to pay them, such as through gift cards or non-financial rewards.

04 Offer a spoonful of sugar:

In some instances, friction might be inevitable despite our best efforts to streamline processes and optimize experiences. Where we can't fix the friction itself, we can offer incentives to encourage people to power on regardless. For example, offer one-on-one Q&A sessions with the development team for people who join your beta program.

Friction at each stage

Let's look at some examples of friction a DevRel program might need to overcome.

Stage	Friction point	Potential solutions
Discover 01	A better-funded competitor gets all the attention	Partner with a more popular DevRel program Place content on channels where your target developers hang out Invest in events where you can meet your target developers
Activate 02	Your company has a reputation for being developer unfriendly	Pay influencers whose work you admire and align with your product vision to cover your product Publicly applaud developers doing awesome things with your product through success stories
Implement 03	Documentation is hard to follow	Create video tutorials Off regular Q&A office hours and live streams Create sample and demo apps Create user observation sessions to provide evidence to the tech writing team
Advocate 04	Most active developers are also too busy to speak at events	Find online speaking opportunities that remove the need for travel time Write slide decks on their behalf to minimize preparation

DevRel flywheel outputs

Outputs from the flywheel are the measurable results of the energy you've put into it. Moreover, they also loop back as inputs, further fueling the ongoing cycle of engagement and growth. This gives the flywheel a two-way relationship with your strategic goals.

Once your DevRel flywheel gains momentum, you'll start to see outputs that contribute to achieving your strategic goals. But it also helps you to define those goals and the KPIs that help you keep on track. Once you drill down into how each stage feeds into the next, the flywheel can give you a clearer picture of what you should be doing.

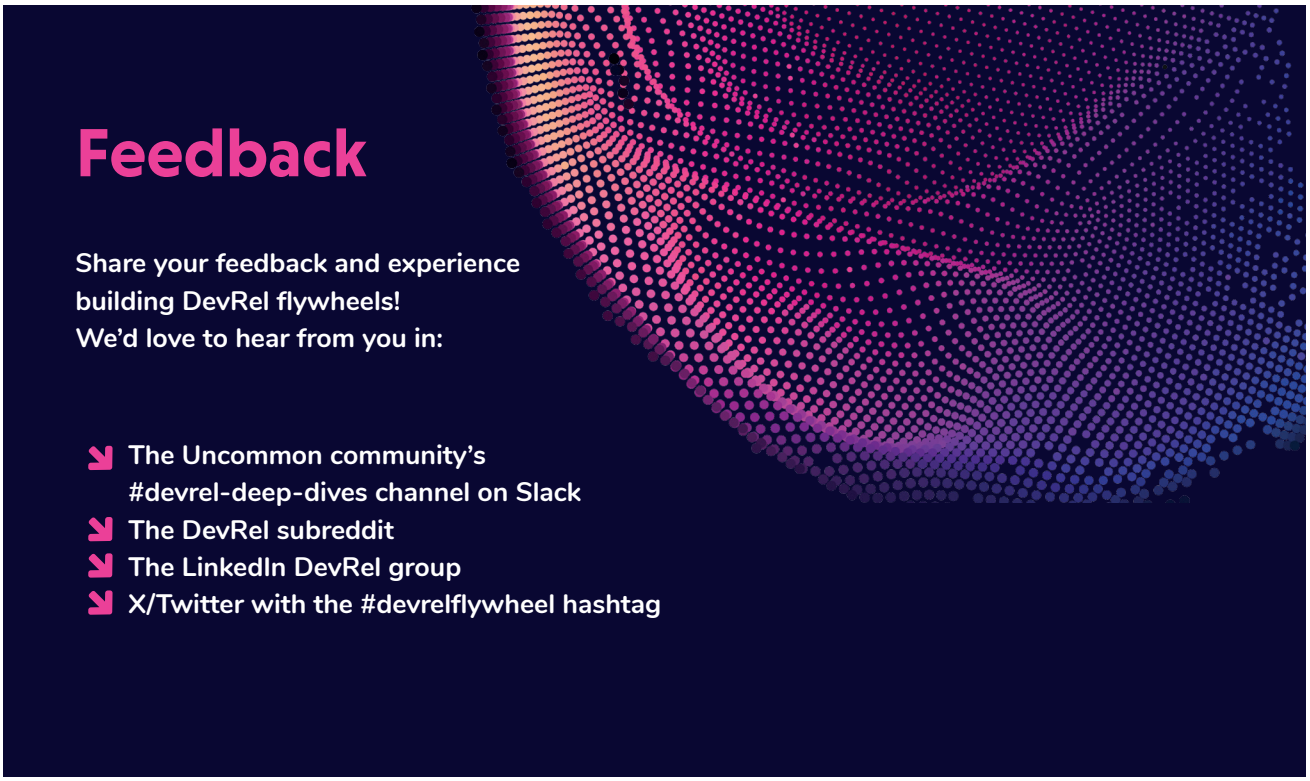
Stage	Outputs
Discover 01	Increased brand awareness among developers Traffic to your website, repos, social accounts, community channels, and videos/livestreams Third-party content written about your product
Activate 02	Increased rate of sign-ups Proof-of-concept projects leading to demo apps Activity on your developer forums, in your repos, in your community, and/or across ecosystem channels like Reddit and Stack Overflow
Implement 03	Developer success stories Higher usage numbers and revenue Efficient identification of onboarding and developer experience issues
Advocate 04	Content produced by people outside your company Improved developer experience thanks to contributions back to the product, such as SDK improvements and open-source contributions Increased awareness in new regions, thanks to the community's wider reach

Closing thoughts

Flywheels are a useful way to think about the work that we do in developer relations. By understanding how each part of our DevRel strategy can feed into the next, we can create programs that amplify our own efforts and take on a momentum of their own.

Here we've provided an overview of how the DevRel flywheel can help you to build an engine of developer adoption. For help improving the efficiency of your DevRel program, check out Common Room.

Now we'd love to hear from you about how you've implemented flywheels in your DevRel and developer marketing programs.



hoopy

The **developer tools** marketing agency

We can help you build and execute DevRel and developer marketing strategies. Talk to us about content, training, strategy, audits, mentoring, and more.

Talk to us: hello@hoopy.io

hoopy.io