The need to standardize DevRel in the enterprise

Tuesday, February 18th, 2025

Introduction

Developer Relations (DevRel) must be a strategic business priority to any organization with a sizeable contingent of developers. As organizations scale, ensuring software developers are aligned with business objectives is critical. Yet explicit functions intended to span the boundaries between leadership and development teams either don't exist or lack standardization.

Internal DevRel is, relative to its external cousin, a rarer species, poorly defined. Without clear positioning, the function is often left to the mercy of changing business priorities, when what it really needs is a clear definition of its business value.

The value of a strategic DevRel function

Large organizations naturally face challenges in scale and alignment, such as:

1. Business units disconnected from developer roadmaps, sometimes unintentionally impeding innovation.
2. A loss of developer trust in organizational priorities.
3. Software policies produced without developer input, leading to direct loss of productivity, workarounds, and quiet non-compliance from developers who just want to get work done.

A well-positioned DevRel function directly addresses these systemic issues by rebuilding broken trust lines between leadership and developers, baking 2-way input loops into policies that affect developer ways of working, and functioning as liasons between business units.

To remain strategically valuable, DevRel must also stop engaging in low-impact activities. A well-rounded DevRel team is a high-leverage function, and its resources should be directed toward initiatives that maximize business and developer outcomes.

What internal DevRel is not

DevRel is not developer marketing or community management

Marketing and community engagement are best handled by dedicated teams. While DevRel may contribute to developer engagement strategies, it does so from the perspective of advocacy, not promotion.

DevRel is not Developer Experience (DevEx)

GitLab defines a developer advocate as "a liaison between developers and the DevEx team, gathering feedback and advocating for developer needs." While this definition is partially correct, it is too narrow. DevRel is not just a bridge to DevEx—it represents the interests of developers across the entire organization.

Developer advocates in the enterprise have a broader organizational remit. They maintain deep knowledge of both business strategy and developer pain points, making them the ideal translation layer between leadership and developers. Their role extends beyond tooling and workflows, influencing cultural, procedural, and policy-level decisions that impact the entire developer ecosystem.

DevRel from first principles

The word "advocate" originates from the Latin *advocatus*, meaning "to call to one's aid." In Middle English, it evolved to mean "one who intercedes for another" or "protector, champion, patron." This is the foundation of our industry standard. DevRel exists to serve and protect developers, ensuring their needs are represented at all levels of the organization.

DevRel has a monopoly on developer trust

Developer trust is the foundational element of DevRel. If DevRel breaks this trust—such as by becoming a one-way conduit for top-down messaging that disregards developer interests—it ceases to be valuable. Effective DevRel operates bidirectionally, gathering insights from developers and ensuring leadership decisions are informed by ground-level realities. When developers are included in decision-making, organizations benefit from policies that align with technical realities and minimize disruption to productivity.

The value of DevRel (at length)

A holistic DevRel practice

A well-functioning DevRel practice recognizes that developer productivity is shaped by more than just tooling. Developers operate within *sociotechnical*[1] systems—interdependent networks of people, technology, and organizational structures.

A strategic DevRel function considers:

1. The cultural and organizational factors affecting developer effectiveness.
2. The policies and processes that impact how developers work.
3. The psychological and cognitive aspects of developer experience.

An effective internal DevRel function looks for root causes of dysfunction at the level of the entire system and applies treatment. Such a function can only be effective if empowered by the very top of any given enterprise, because of its systems-level actions.

An example: a new acquisition involving a hundred-plus developers naive to the organization necessitates cultural and technical onboarding. Do the new developers feel psychologically safe enough to contribute to a codebase? Do they have the developer tools required to contribute? How do they gain access to cloud resources? What best practices must they know before contributing?

All of these questions are best solved by a function both technically proficient and organizationally savvy. That's DevRel.

DevRel as boundary spanners

> "Boundary spanners facilitate the sharing of expertise by linking two or more groups of people separated by hierarchy, location, or function."—Tamara Keszey

1. Sociotechnical systems and boundary spanning roles were introduced to me by Dr. Jabe Bloom, a genius in organization theory. You should drop everything you're doing to watch his talk, Whole Work: Sociotechnicity and DevOps (https://www.youtube.com/watch?v=WtfncGAeXWU) and then come back to this article after.

DevRel acts as a critical boundary-spanning function within an organization. Consider security and compliance. When new software policies are created by parts of an organization without developer input, those policies can freeze developer productivity in place and shatter trust.

Boundary spanners do the messy work of bridging two worlds, often speaking two different languages. In the example of software policy, these teams often do not possess their own developers and may make assumptions about impact and architecture unmoored from technical consequence. DevRel, external or internal, has *always* done this sort of bridging, and is uniquely suited to this diplomatic task.

DevRel is "user 0", and the ideal go-between to facilitate collaboration between security teams and developers. A DevRel function might run a limited test of new software policies internally and provide feedback or run empathy sessions[2] with other developers.

Translation layer between leadership and developers

A core function of internal DevRel is translating between developers and leadership. Developer advocates are in touch with more developers, more often than anyone in your business. The fact they know more about how each developer team is contributing to your business outcomes than anyone else makes them the ideal translation layer between leadership and developers.

If enterprise leaders are put in charge of initiatives with the power to make sweeping impact to how developers deliver software, DevRel is an essential consultant to those leaders.

Conclusion

Internal DevRel is a strategic function that, when positioned effectively, can bridge the gap between developers and business leadership. Organizations that invest in DevRel as a core function—rather than an auxiliary role, or worse an enterprise mirror of external DevRel activities—harvest developer trust, create and sustain high-performing and empowered delivery teams, and align business priorities to technical realities.

2. https://mixtape.swyx.io/episodes/empathy-and-the-hard-way