The platform engineering community just published Volume 3 of the State of Platform Engineering report and a key question that was asked to platform teams was which tools they prefer using for their platform journey, across different parts of the toolchain.

These data points, combined with 100s of PlatformCon talks and the thousands of threads in the Platform Engineering Slack, where practitioners share their challenges and best practices around platform tech stacks, can give us a unique look into what tools are trending in the industry.

Platform teams that don't think holistically about their Internal Developer Platforms (IDPs) and don't make sure their platforms cover all bases across the different tooling categories will fall behind. That's why we decided to pull together the top 10 platform engineering tools you should watch in 2025 to ensure your IDP is ahead of the curve.

There's no better way of thinking through these categories than following the well-established community blueprints for enterprise IDPs. We'll drill down into the five platform planes and double-click on some of the most important sub-verticals within those. Let's dive in.

Platform Orchestrators: Humanitec

Building an Internal Developer Platform is like building an application, you should start from the backend and define the business logic of your platform, then make your up to frontend interfaces like portals, and add security and observability solutions later.

There are two main design patterns for platform backends: pipeline-based and graph-based. Graph-based backends, or Platform Orchestrators, are designed to handle complex logic and all sorts of enterprise architectures. Check out our pipeline vs graph backend article for more infos.

When it comes to Platform Orchestrators, the clear leader is the Humanitec Platform Orchestrator. It was the first of its kind and kickstarted the whole category. It's been mentioned in 9 Gartner Hype Cycles in 2024 (e.g. Platform Engineering, Cloud Computing , SRE or Infrastructure and Operations) and is the established solution for graph-based platform backends across many enterprises and Fortune 500s.

The Platform Orchestrator is designed to shield application developers from the complexity of the underlying toolchain. It lets them describe what they need in an abstract way (e.g. my workload needs a Postgres), without having to worry about any implementation details. The Orchestrator reads in this declarative, abstract request and matches it to the rules and templates defined by the platform team. It then creates a graph-based representation of the workload and its dependencies, outputting executable configuration files (e.g. Helm charts, Terraform modules) that can be deployed directly by the Orchestrator or by an existing CD solution (e.g. synced to a cluster by ArgoCD).

There have been plenty of talks at PlatformCon24 covering the Humanitec Platform Orchestrator (e.g. from RedHat, Bechtle or Convera).

CI: GitHub Actions

It's important to mention that opting for a graph-based backend over a pipeline-based one doesn't mean that you no longer need your CI/CD toolchain. It's simply a matter of using CI/CD tooling for

what they are designed to do (e.g. environment progression logic) and rely on Platform Orchestrator for complex logic, RBAC, compliance flows and other enterprise features your IDP needs.

When it comes to CI, there's a clear winner since GitHub started growing from just being a Version Control System (VCS) into CI/CD back in 2018. GitHub Actions has the huge advantage of being native to the GitHub ecosystem, with a seamless integration with their VCS. As GitHub is the market leader in VCS, for most teams it just makes sense to go with their CI pipeline offering.

Of course, there are alternatives: You can similarly use GitLab, covering the entire SDLC from VCS to CI/CD, or you can opt for specialized CI tools like CircleCI, Jenkins, or Travis CI.

CD: ArgoCD

Moving on for a quick look over in the CD category, ArgoCD emerged as the preferred solution by platform teams, especially when it comes to implementing GitOps-first setups. ArgoCD allows for a wide range of deployment setups, such as air-gapped environments for high-security use cases. See this guide for building secure Internal Developer Platforms in regulated environments by André Alfter at PlatformCon24.

However, if you are working in less regulated environments or you do not want to follow the GitOps path, tools like GitHub Actions or GitLab might be good enough.

Developer Portals: Backstage

Once you defined your platform business logic and designed golden paths for your developers to follow in your platform backend, you can move up a layer and pick the interface your developers will use to consume such templates and paved roads.

Here you have a few options: code-based (see Workload Specifications below), API, CLI, or UI-based interfaces. UI-based interfaces to access your Internal Developer Platform are also known as Developer Portals, and here the clear leader is the solution open-sourced by Spotify: Backstage.

Backstage has by far the largest ecosystem and market share, with the vast majority of platform teams still opting for it. While it gives organizations the highest degree of flexibility, which is why the majority of enterprises choose it, it should be said that the implementation requires advanced engineering skills.

That's where closed-source commercial offerings are gaining some traction. A good example is Port, which launched in 2019 and recently raised $60M in Series C funding in 2024. Offerings like Port provide a much easier and faster onboarding for platform teams, despite being less customizable. Port comes with classic portal features like service catalog and scaffolding and, similar to Backstage, has a good ecosystem of integrations (e.g. integration with Humanitec).

Workload Specifications: Score

An alternative (or complement) to UI-based interfaces for your IDP are code-based interfaces like Workload Specifications. From Thoughtworks Tech Radar:

“Many organizations that implement their own Internal Developer Platforms tend to create their own platform orchestration systems to enforce organizational standards among developers and their platform hosting teams. However, the basic features of a paved-road deployment platform for hosting container workloads in a safe, consistent and compliant manner are similar from one organization to another. Wouldn't it be nice if we had a shared language for specifying those requirements? Score is showing some promise of becoming a standard in this space. It’s a declarative language in the form of YAML that describes how a containerized workload should be deployed and which specific services and parameters it will need to run.”

In 2024 Score was taken under custodianship by the Cloud Native Computing Foundation (CNCF) as an open-source project. With the backing of the CNCF, Score has the potential to be widely used beyond the two reference implementations (Kubernetes and Docker Compose) that it was originally released with. The extensibility of Score will hopefully lead to community contributions for other platforms. Score certainly bears a resemblance to the Open Application Model (OAM) specification for Kubevela, but it's more focused on the deployment of a single service, rather than the entire application. It’s certainly off to a very strong start, with over 7.8k stars, 2.2k forks on https://github.com/score-spec.

To see some examples of Score in action check out Paul Revello’s talk on how to use Score for your IDP built on GCP (using the Google Hipster Store) or Mathieu Benoit’s, CNCF Ambassador and Score contributor, on crafting and improving developer experience with Dapr+Score.

Cloud Development Environments (CDEs): Gitpod

Jumping over into another category of our Developer Control Plane (the frontends and interfaces of your IDP), Cloud Development Environments, or CDEs, help teams drive standardization by providing them with pre-installed tools and dependencies to get started coding.

According to Gartner, CDEs increase developer productivity, while improving security and supporting compliance. They have recently been mentioned in the Platform Engineering Hype Cycle and are the subject of a Market Guide.

Gitpod is one of the leading vendors here, enabling standardized development environments that run both on your laptop and in the cloud. Gitpod is used by over 1.5 million developers including some of the largest organizations in the world.

Infrastructure as Code (IaC): Terraform

A quick look also at the last vertical in our Developer Control Plane: Infrastructure as Code, or IaC. No surprises here, with Terraform still number one according to the data from the latest State of Platform Engineering report.
IaC setups serve as the source code of Internal Developer Platforms, allowing platform teams to automate the provisioning and management of infrastructure. If implemented well and paired with a platform orchestration tool, IaC ensures consistency and reduces manual errors, making deployments more reliable.

Terraform leads this category, with 71% of respondents to the State of Platform Engineering Vol 3 survey choosing it for its modular cloud-agnostic features that streamline infrastructure management. Crossplane follows with 13%, while OpenTofu, adopted by 7%, stands out as the open-

source alternative to Terraform, providing similar functionality without vendor lock-in. Finally, Pulumi accounts for only 1% of teams surveyed.

Database as a Service (DBaaS): Aiven

Aiven was launched in 2016 and has since become a leader in the Database-as-a-Service (DBaaS) category, particularly for its open-source approach. Aiven offers flexibility with its support for multiple databases, including PostgreSQL, Apache Kafka, Redis, and Cassandra. Its platform simplifies cloud database management, enabling businesses to set up and maintain databases efficiently without being locked into a specific vendor. Aiven's ability to integrate seamlessly with existing cloud infrastructures makes it an attractive option for organizations aiming to maintain agility and scalability in their operations.

Monitoring and Observability: Grafana and Prometheus

Prometheus and Grafana together form a powerhouse for monitoring and observability in platform engineering. Prometheus, a CNCF-graduated project with over 55k stars on GitHub, is widely regarded as the leader in real-time metrics collection and monitoring. Initially launched by SoundCloud in 2012, Prometheus has become a staple for platform teams, particularly in Kubernetes environments, where its flexibility and efficiency in handling time-series data are unparalleled.

54.59% of those surveyed in the State of Platform Engineering report mention Prometheus, reflecting its strong adoption in enterprise and startup contexts. Its powerful PromQL (Prometheus Query Language) allows developers to define precise, actionable alerts and visualizations. The tool's dimensional data models offer rich insights, while built-in sharding and federation features ensure scalability for large-scale deployments. Prometheus also integrates seamlessly with a wide range of third-party data sources, making it a versatile choice for diverse tech stacks.

Grafana serves as an intuitive visualization layer that complements Prometheus, enabling teams to create real-time dashboards to monitor performance metrics, identify bottlenecks, and gain actionable insights. Its highly customizable interface and extensive plugin ecosystem make it the preferred choice for building comprehensive observability solutions. Grafana supports multiple data sources beyond Prometheus, adding flexibility for broader monitoring needs.

Survey data highlights Prometheus as the clear leader in the monitoring space, with strong competition from Dynatrace, Datadog, and the ELK Stack, which remains popular for log management. Together, Prometheus and Grafana offer platform teams a robust observability stack that balances the flexibility of open-source tools with the advanced capabilities needed for enterprise-level monitoring.

Security Suite: Snyk

Snyk is recognized as a leader in its category in the State of Platform Engineering Vol. 3 report. Its proactive approach to securing the application lifecycle - from code to deployment - sets it apart. According to the report, Snyk is trusted by a majority of platform teams for its robust capabilities in identifying vulnerabilities across Infrastructure as Code (IaC), open-source dependencies, container images, and more.

In addition, Snyk's developer-centric design ensures ease of adoption, empowering teams to fix vulnerabilities without needing deep security expertise. These features make Snyk a cornerstone tool for secure Internal Developer Platforms.

Summary

This article hopefully gave you a good overview of the different categories of tooling for your Internal Developer Platform and what top-performing organizations currently prefer to use in each vertical. Beyond the individual tool recommendations, it's important that as a platform team, you make sure you are covering all the key components of your platform engineering toolchain. Doing so by following best practice architectural patterns, like starting from your platform backend, will be the determining factor in how successful your IDP will be.

If you'd like to learn more about the latest trends in tooling and how to leverage established blueprints like IDP reference architectures and the Minimum Viable Platform framework, check out our Platform Engineering courses and certifications.