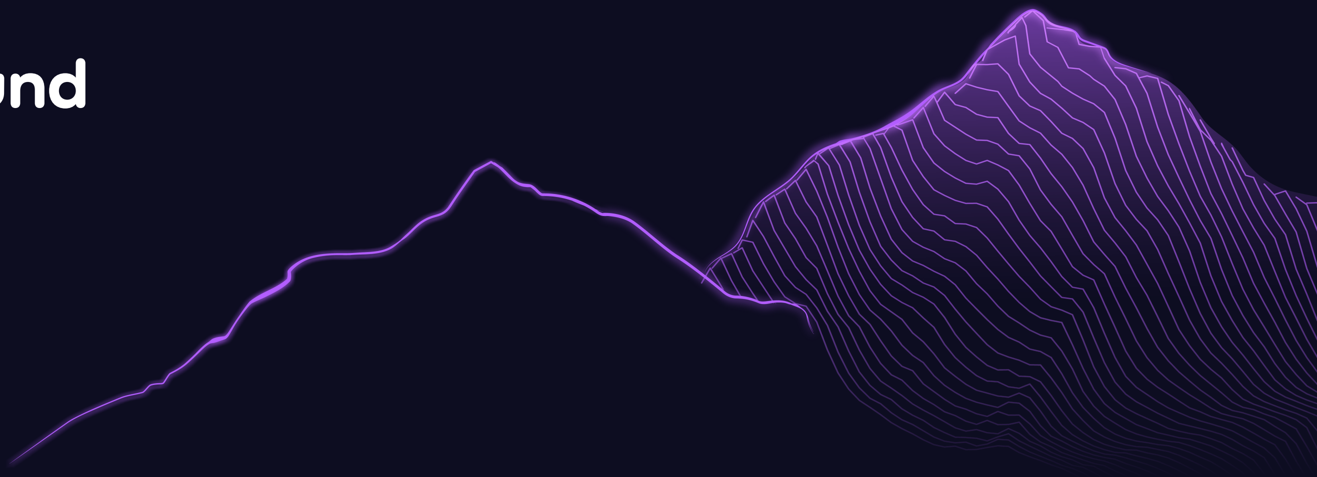
upbound



Optimizing Terraform with Crossplane

Terraform has served as the industry standard to automate infrastructure provisioning. However, with the rise of Kubernetes and Crossplane, it's time to bring infrastructure-as-code to a control planes based architecture.

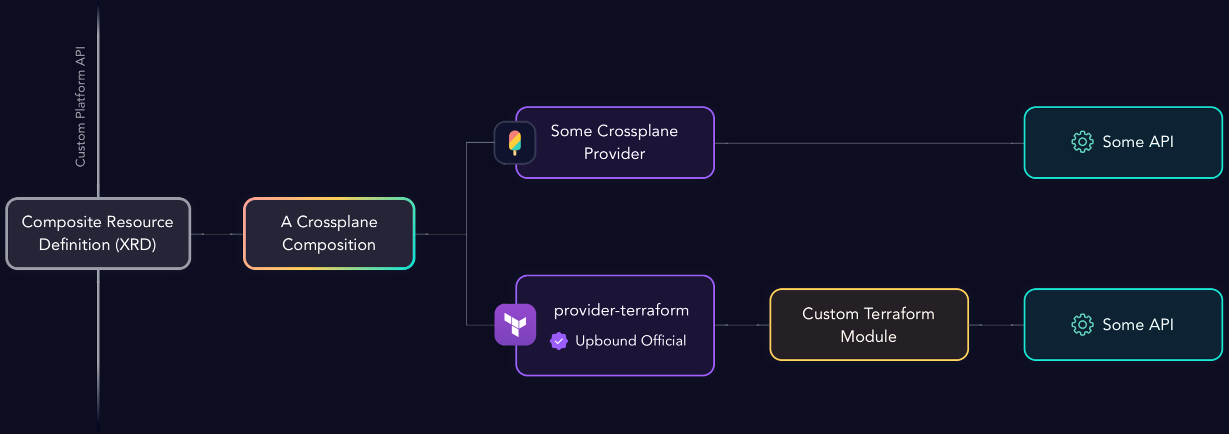
WHY CROSSPLANE?

1. Align the infrastructure dependencies lifecycle with app lifecycles: with Crossplane, you can deploy an application definition and its infrastructure dependencies in the same deployable unit
2. Mitigate configuration drift without additional investment
3. Enforce IT policies using OPA or Kyverno to automatically ensure continuous compliance
4. Secure existing investments by supporting integration scenarios

DO YOU HAVE TO RIP AND REPLACE?

Absolutely not. It's understandable to want to take a phased approach rather than migrating all at once due to reasons such as high investment in custom domain-specific Terraform logic. With the new Official Crossplane Terraform Provider, you can make a control plane out of your existing Terraform code and extend it with the native Crossplane power.

HOW DOES IT WORK?

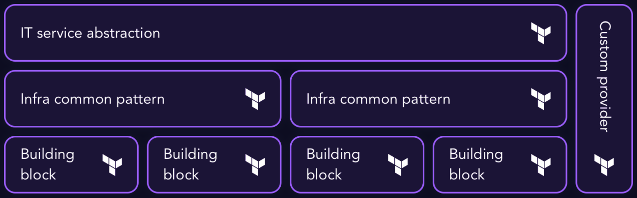


A typical setup where provider-terraform includes a custom Terraform module and combines it with some other Crossplane provider into a custom Crossplane composition.

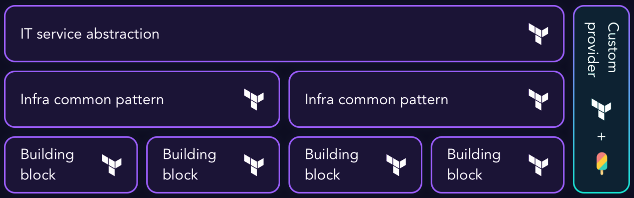
CHANGING THE WHEELS WITHOUT STOPPING – MILLENNIUM BCP

Millennium bcp is Portugal's largest private-sector bank, with 1,334 branches serving more than 6 million customers around the world. The key to its vision and success has been continued investment in the latest technologies to generate business value. They decided to implement Crossplane to tackle some of the issues their Terraform infrastructure created through a multistage approach.

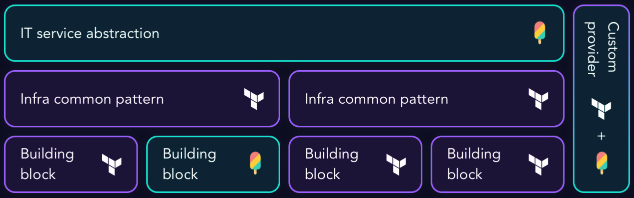
Step 0: Existing Barebone Managed Resource of Terraform Workspace



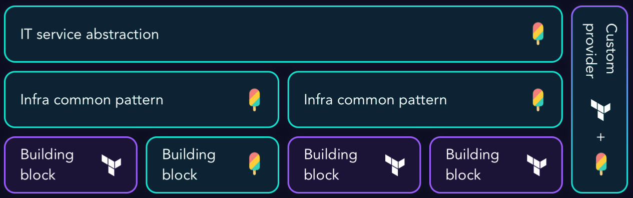
Step 1: Build and include Custom Crossplane Provider



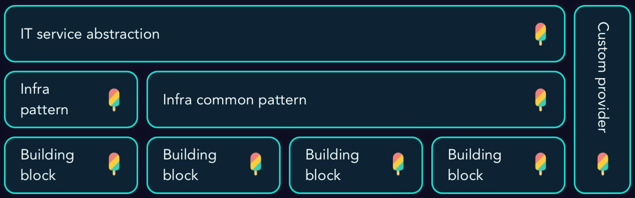
Step 2: Migrate the Service Abstraction while continuing to use common patterns and most building blocks



Step 3: Migrate common patterns to a mixed scenario of Terraform Workspace and Crossplane-native Resource within the Composition



Step 4: Migrate fully to Crossplane (Optional)



As a result, they found

Problems Faced with Pure Terraform

- Unrecoverable State Drift
- Unmanageable Compliance
- Internal customers were not seeing the same efficiency gains as infrastructure teams
- Infrastructure dependency patterns not an integral part of application patterns

Result with Crossplane

- Reduced human error in provisioning resources
- Improved compliance throughout the resource's lifecycle
- **Went from SLA of 8 days for project and dependencies set up to just a few minutes**
- Greater visibility on application dependencies and health

The inventors of Crossplane

Upbound invented the popular open-source project, Crossplane. With over 7,500 Github stars and over 8,000 Slack members, Crossplane is a framework for building cloud native control planes without needing to write code. Upbound democratized control plane technology with the Crossplane project, and continues to drive it forward as maintainers and contributors.

Learn how to start migrating your platform over today: