Nine Questions to Build Great Infrastructure Pipelines

Rob Hirschfeld
@zehicle
CEO & Co-Founder
RackN

Photo by Kateryna Babaieva from Pexels
Hard Lessons

RackN has been helping companies deploy massive fleets of global infrastructure.

Self-Managed by the Customer
Not a SaaS or Consulting
Hard Questions

RackN has been helping companies deploy massive fleets of global infrastructure.

Self-Managed by the Customer
Not a SaaS or Consulting

Why...

...is automation never enough?
...do we write do we keep writing custom automation?
...does our automation age so badly?
...can’t we know if we’ve automated “correctly”?
Because Our Environments are Complex & Siloed Infrastructure Automation Tools

Service Orchestration Tools

Clustering Tools
- Applications

Configuration Tools
- Platforms

Provisioning Tools
- Operating Systems
- Physical Infrastructure (Compute/Net/Store)
And Complexity Grows as We Add Abstractions

- Physical Infrastructure (Compute/Net/Store)
- Operating Systems
- Platforms
- Applications
- Virtualization
- Clustering Tools
- Configuration Tools
- Provisioning Tools
- Service Orchestration Tools
Wouldn’t it be great if we could make automation that works across layers?
Infrastructure Pipelines

We need to start looking at automation as a connected sequence of modular operations with state, configuration and process.

So, what questions should we be asking to guide us towards this type of system level, reusable automation?

Photo by Benjamin White from Pexels [link]
We need to start looking at automation as a connected sequence of modular operations with state, configuration and process.

So, what questions should we be asking to guide us towards this type of system level, reusable automation?

Let’s start with these nine...
1. Why doesn’t my CI/CD pipeline understand infrastructure?

CI/CD pipelines flow artifacts, not resources. Integrating them makes sense, the language or resource consumption and state is different. We need pipelines that understand environmental constraints, resource life-cycle, drifting configuration, cluster join/drain operations.

*It’s a matter of continuous cycles vs linear process.*
2. Why is a “Pipeline Flow” different than other Orchestration

Pipelines focus on a destination while Orchestration focuses on coordination and path.

Pipelines are *mostly* linear sequences which drive to a clear target state or goal. While Orchestration builds a “if-then” map to find the correct outcome.

* RackN pipelines do have dynamic elements to accommodate environmental changes.
3. Why focus on Intents instead of the chained actions?

A Pipeline’s **Intent** provides us with a critical abstraction target for APIs.

Pipelines can hide the complexity of infrastructure types, constraints, prerequisites and other challenges.

Since consumers of the pipeline don’t worry about what goes on inside the pipeline, **they can focus on the outcome.**
4. Why are provisioning and configuration so different?

To build an end-to-end Infrastructure Pipeline, we have to mix different types of automation. Provisioning, Configuration, Monitoring, and Cluster Coordination are required but different.

Each type has a different operational context. The scope, authority and location of pipelined automation must constantly change!

Photo by winrood lee from Pexels
5. Why can’t I share state between tools?

**Ops tools are built bottom up** to excel in their context; they don’t worry about sharing state.

This means multiple “single source of truth” tools are always in conflict. But... In infrastructure state is critical and constantly being changed by many different actors.

Pipelines require a standardized way to **share and synchronize state** even with isolated tools.

Photo by Şahin Sezer Dinçer from Pexels
5 of 9!

Nine is A LOT of questions!
6. Why can’t I ignore what is between pipeline flows?

Most (>99%) of a resources’ time is spent outside of its transformation pipeline.

So we still need to manage and automate resources outside of Pipelines because infrastructure is persistent and dynamic.

We recognized that targets have two modes: pipeline mode and service mode. Each mode requires a different automation approach.
7. Why is IaC central to Infrastructure Pipelines?

IaC “but it’s in git” is a long way from modular and reusable automation. **We need IaC to have code-like libraries for common automation.**

Even so, IaC is directionally critical to building and sharing interchangeable automation components. Like how building codes and standards have made it normal to fit together COTS components. **Ultimately, structure accelerates innovation.**
8. Why is it so hard to reuse and share automation parts?

Infrastructure as Code should make it easier to reuse or share automation components!

But, we need to isolate universal, repeatable automation parts from site-specific parts.

By building pipelines with standard pre- and post- operations*, we can finally decouple site-specific automation from common steps.

* call-back, classify, validate, and “ad hoc” tasks
9. What is holding us back from doing this work?!

In DevOps culture, we strive to work across silos and avoid technical debt. But then we keep us building single site or single team automation.

It takes discipline to be part of a community effort and external collaboration to maintain a system.

But shared and curated processes ultimately accelerate us and improve safety. They allow unique value to be added in a maintainable way.
We’ve been collaborating with our enterprise customers to build a “Universal Workflow” in Digital Rebar.

Out of the box, it includes *all* the process steps required to fully deploy any platform in our library.

Operators select a target profile and workflow chooses the right stages to deliver that target.
What is a Standardized Infrastructure Pipeline?

**Infrastructure Pipeline**
- Discovery
- Burnin
- H/W Config
- O/S Install
- App Install

**Service**
- Discovery
- Burnin
- Hardware
- O/S Install
- App Install
What is a Standardized Infrastructure Pipeline?

Infrastructure Pipeline

- Discover
- Pre Notify
- Standard Pre Process
- Segment Work
- Standard Post Process
- Post Notify
- Chain

Service

- Discovery
- Burnin
- H/W Config
- O/S Install
- App Install

Standardized Call Back

Standardized Call Back
What is a Standardized Infrastructure Pipeline?

Infrastructure Pipeline

- Discover
- Pre Notify
- Standard Pre Process
- H/W Config
- O/S Install
- App Install
- Post Notify
- Chain

Segment Work

Standardized Extension Points

- Ad hoc Tasks
- Classify
- Validate

Standardized Extension Points

- Ad hoc Tasks
- Classify
- Validate
Five things that can help you build Infrastructure Pipelines.

1. Modularize your automation
2. Standardize your interconnects
3. Share state between layers
4. Mix Configuration, Provisioning, Monitoring and Orchestration
5. Never assume using abstractions means layers are decoupled

Image: Jimmy Tidwell via Pinterest
Thank you!

We’re sponsors, please come say “hey” to the team that the booth!

Rob Hirschfeld
@zehicle
CEO & Co-Founder