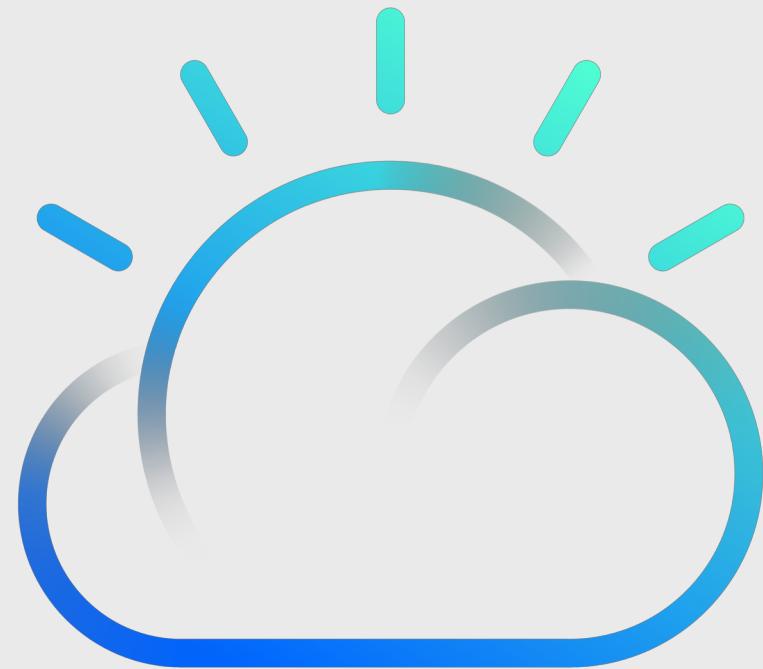
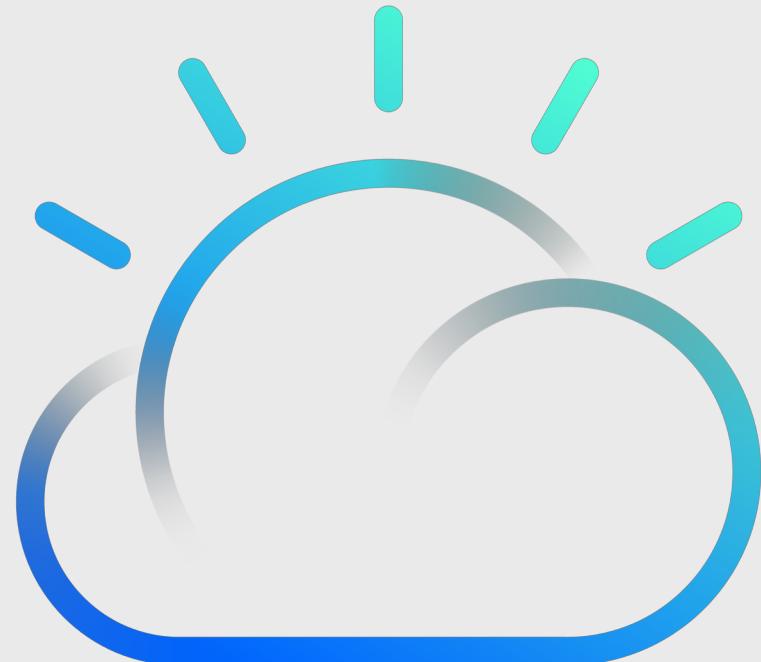


# **Cloud Automation Manager**



**IBM**

**IBM Cloud**



## Level Set

The use cases and market forces, the architectural context within the overall IBM Cloud strategy and the functional components of the solution.

# Use cases driving private cloud adoption

Optimize legacy apps with cloud

Self-service Experience
Next Generation Middleware, Data & Analytics
Automation & Orchestration
Containers & Common Services

Cloud-enabled middleware

Open your datacenter to work with cloud services

APIs
Integration Services & Cloud Native Programming Models
Automation & Orchestration

Integration & Hybrid Cloud

Public Cloud Services
Machine Learning on p/z
Blockchain
Business Process
Data & Apps
On-Premises Software & Services

Create new cloud native applications

New Applications
Cloud Native Services & Runtimes
Automation & Orchestration
Containers & Common Services

New Applications

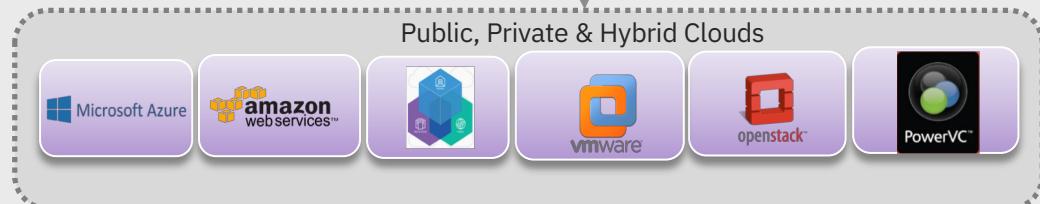
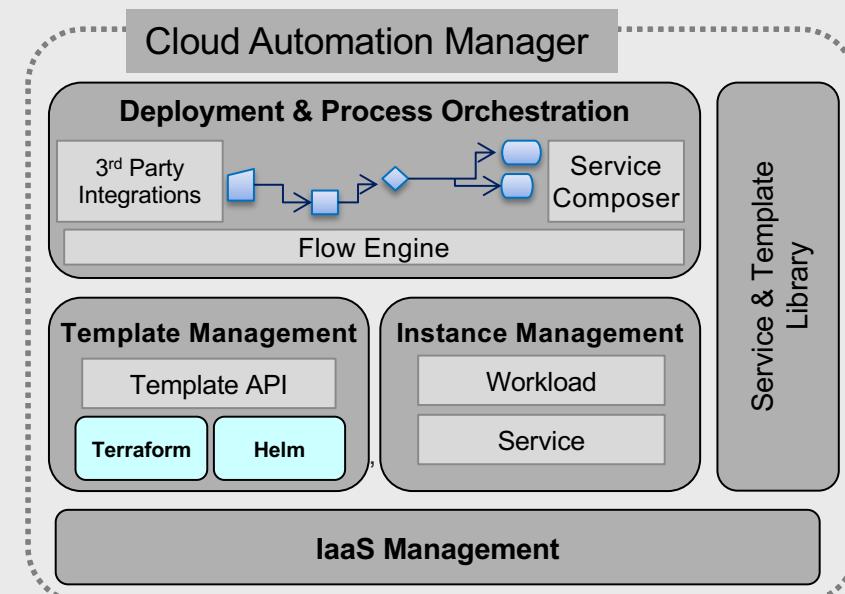
# IBM Cloud Automation Manager: Full stack automation and service orchestration

**Automated provisioning** of infrastructure and applications with workflow orchestration

**Self-service access** to cloud infrastructure and application services

**Manage and govern** workloads across multiple and hybrid clouds

Built with **open technology** to avoid vendor lock-in



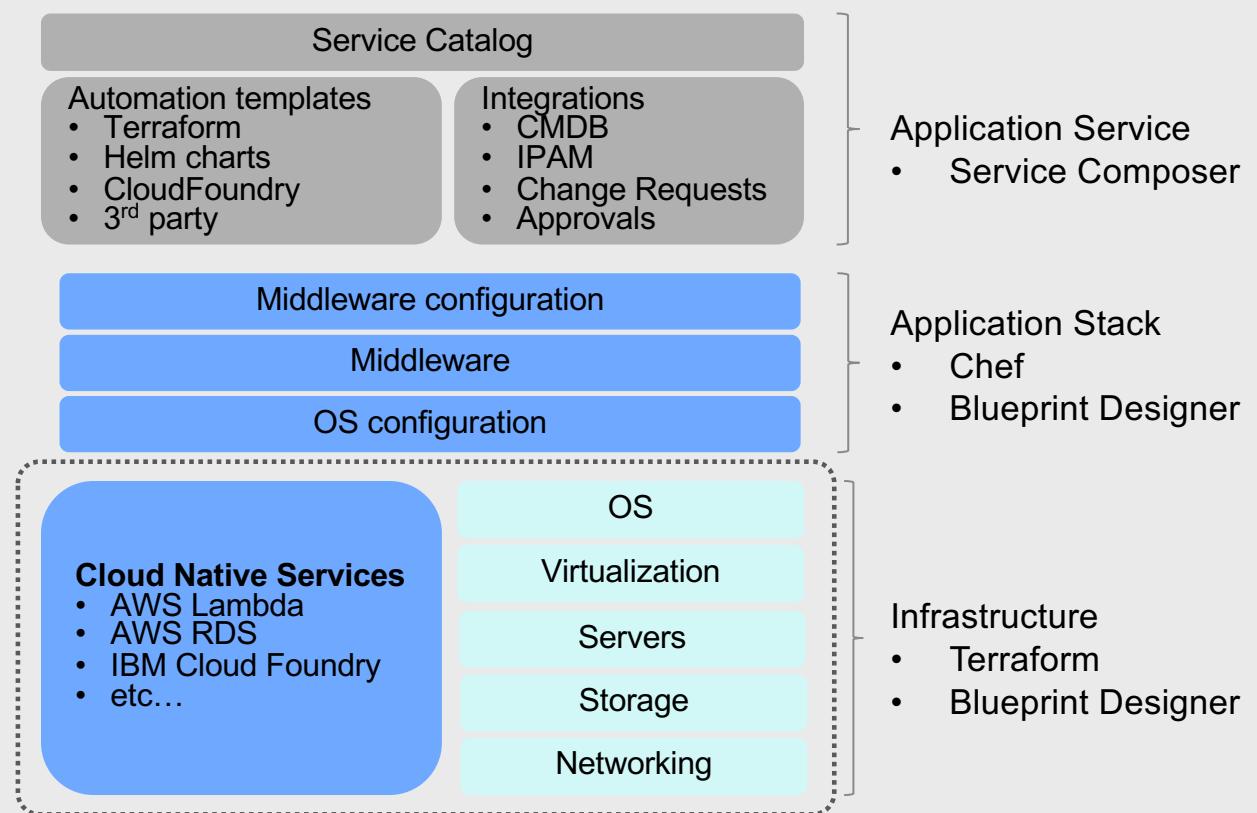
# The Cloud Automation Manager business

All clouds, one tool

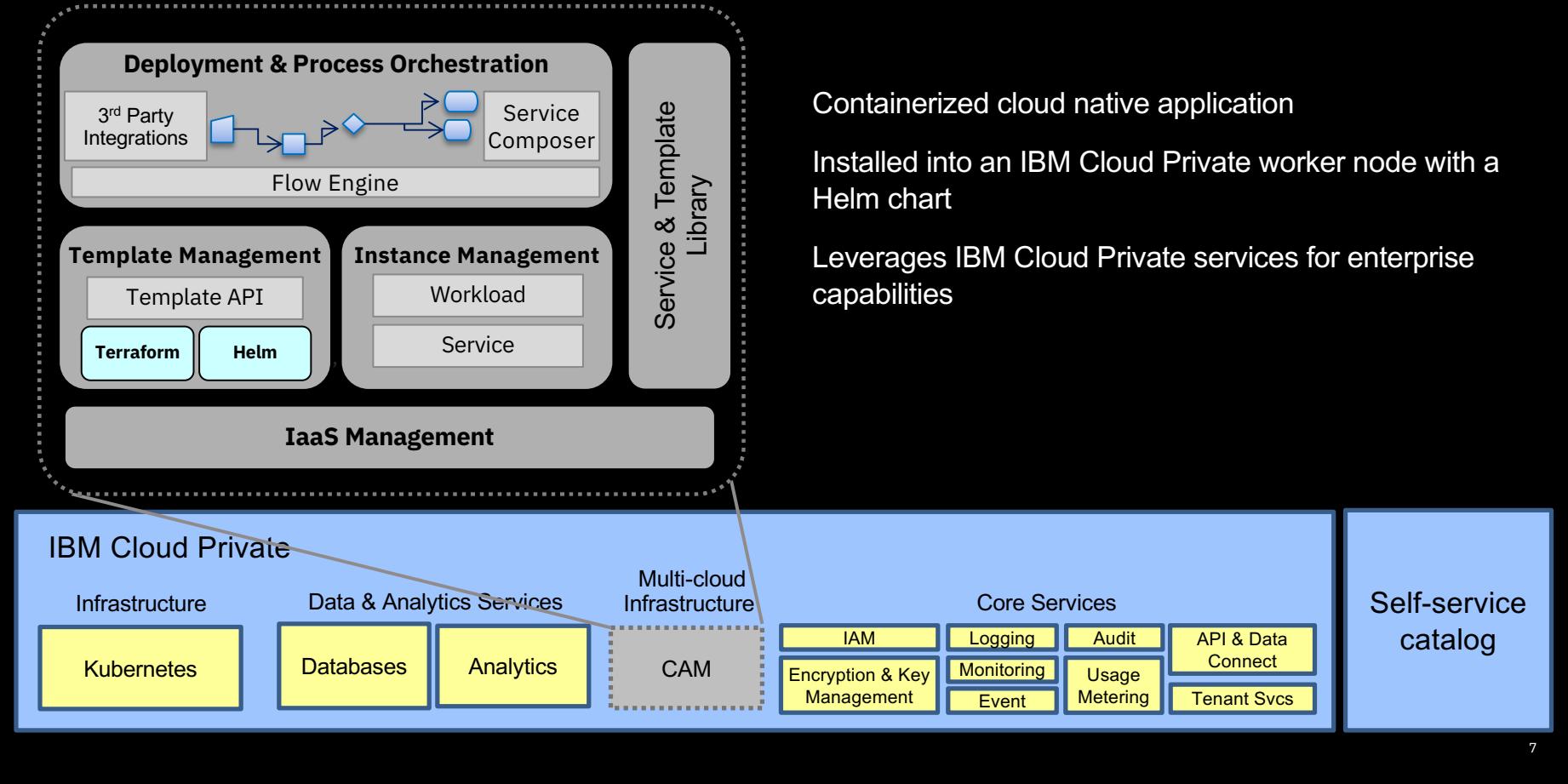
Any workload architecture

Graphical experience

Purpose build for integration

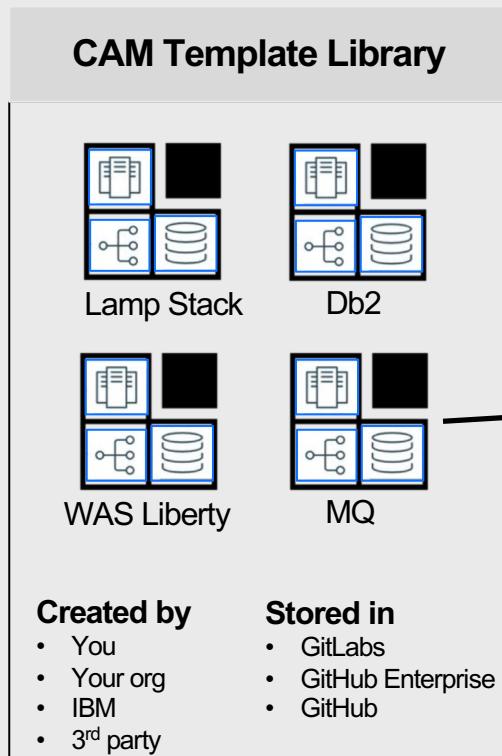


# Cloud Automation Manager in IBM Cloud Private

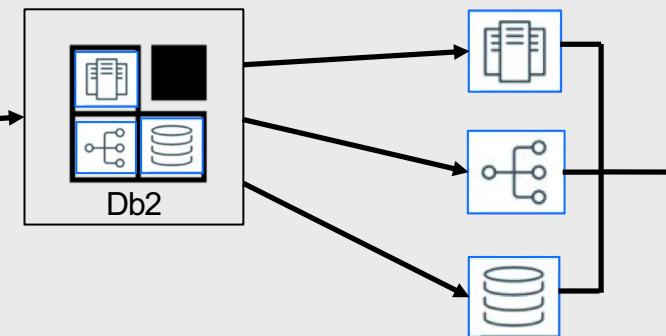


# Using Terraform Configurations

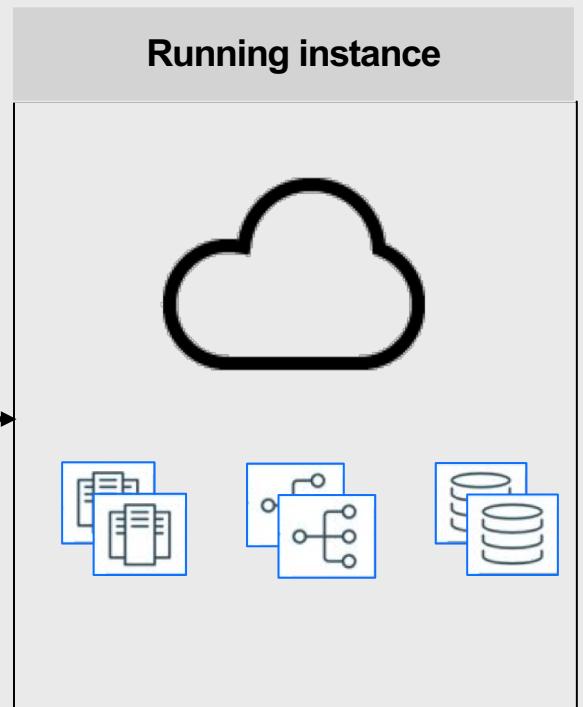
1. Select a template



2. Review and apply plan

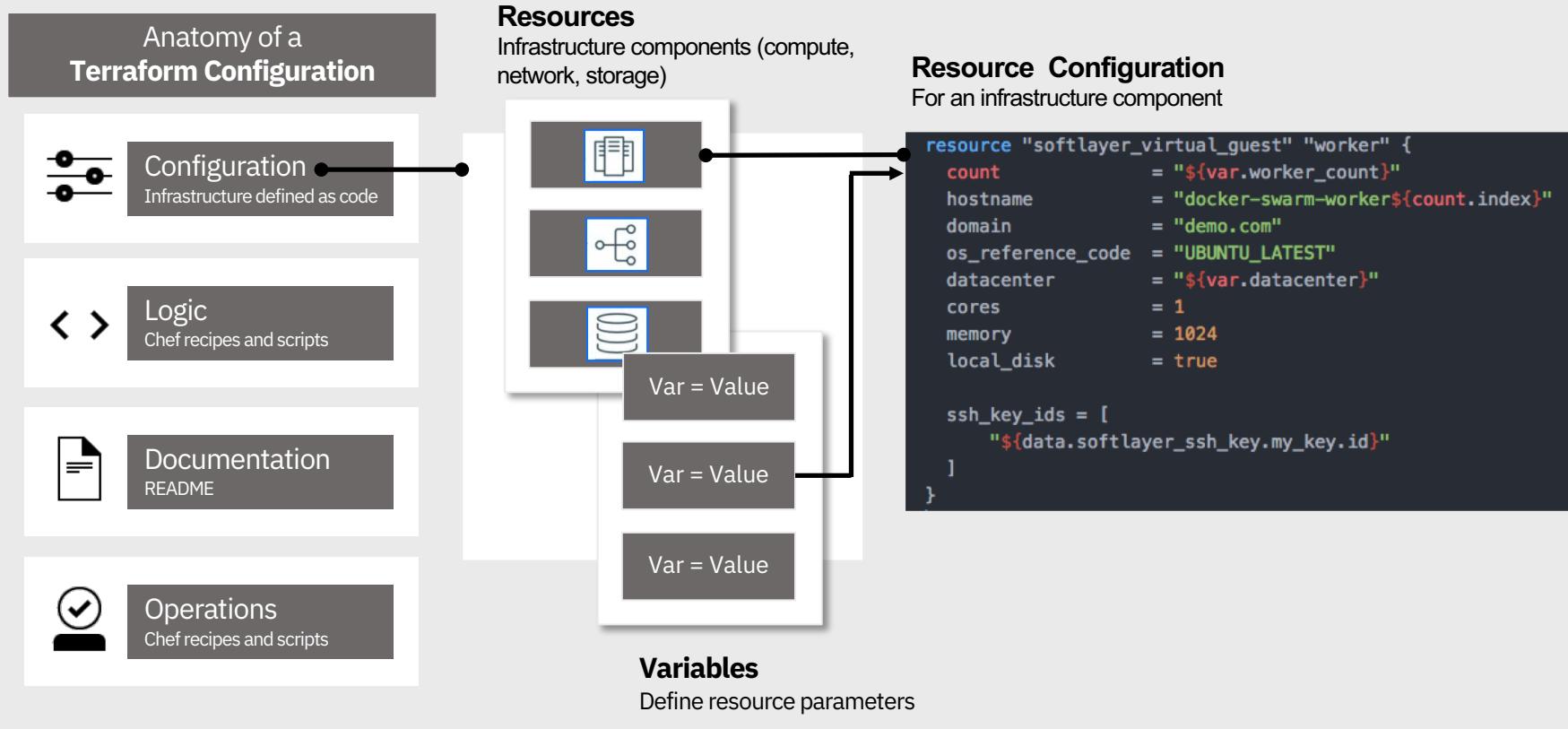


3. Resources are provisioned

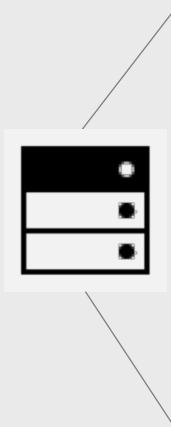


4. Use the environment

# Anatomy of a Terraform Configuration



# Managing cloud infrastructure as code



```
resource "softlayer_virtual_guest" "worker" {
  count          = "${var.worker_count}"
  hostname       = "docker-swarm-worker${count.index}"
  domain         = "demo.com"
  os_reference_code = "UBUNTU_LATEST"
  datacenter     = "${var.datacenter}"
  cores          = 1
  memory         = 1024
  local_disk     = true

  ssh_key_ids = [
    "${data.softlayer_ssh_key.my_key.id}"
  ]
}
```



Store Terraform configuration in Git and manage infrastructure as code

**Accelerate development velocity** with reusable infrastructure based on open source Terraform

**Improve governance and transparency** by tracking the ‘who’, ‘what’ and ‘when’ of all environment changes

**Improve development team collaboration** by enabling team members to easily share application environments

**Reduce configuration drift** by making it easy to track changes to your running environment

# Template lifecycle – Plan/Apply

Allows user to make changes on running instances

Two major scenarios:

1. User want to change a parameter on an instance
  - Example – change memory/cpu
2. User wants to update to a new version of the template used to create their instance
  - Example – add more VMs or other components

Instance modification/update in a two-step process:

1. Plan - allows user to select the version and /or change parameters and execute a phase to see what will change in the environment
2. Apply – executes the changes presented on plan action on the environment

# Creating Terraform templates – CAM Template Designer

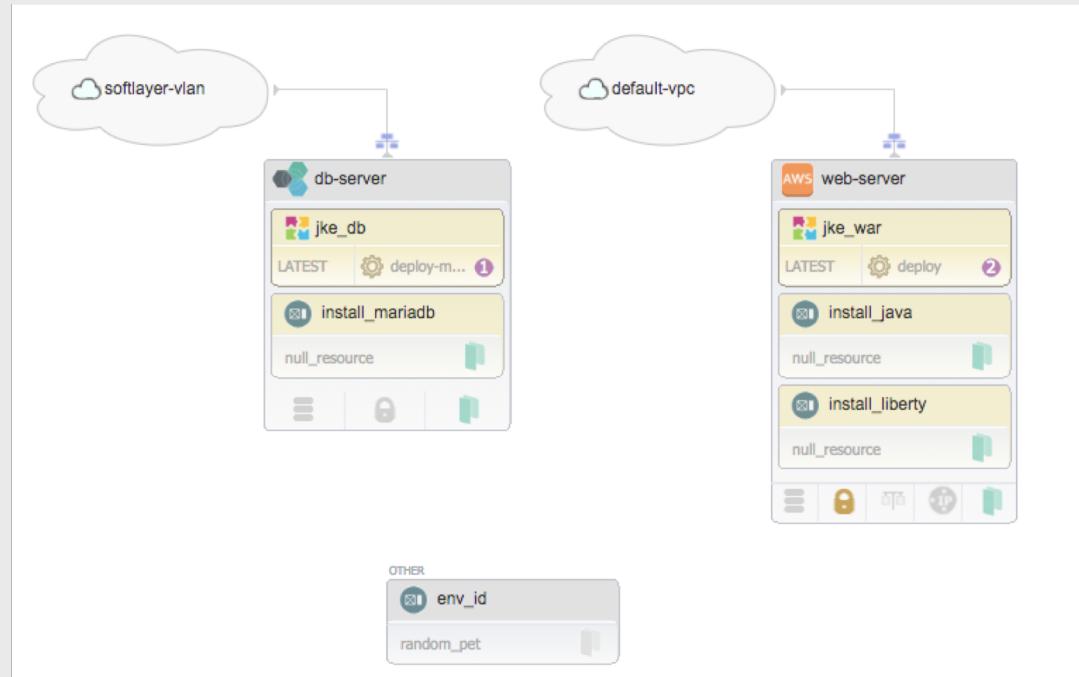
Graphical and text-based Terraform template designer

Built-in integration with Git to easily share and re-use templates

Rich cloud support to create complex Terraform content via drag and drop

Easily create and publish CAM templates

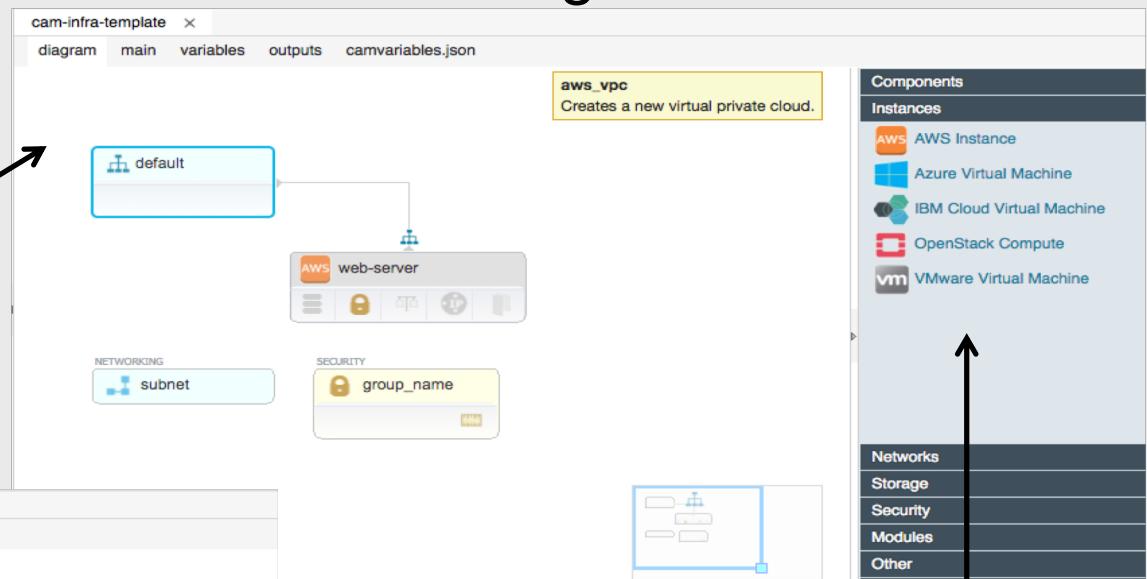
Integration with UrbanCode Deploy to automate full-stack application deployments



# CAM Template Designer – Terraform Design

Graphically create Terraform templates while leveraging Terraform constructs.

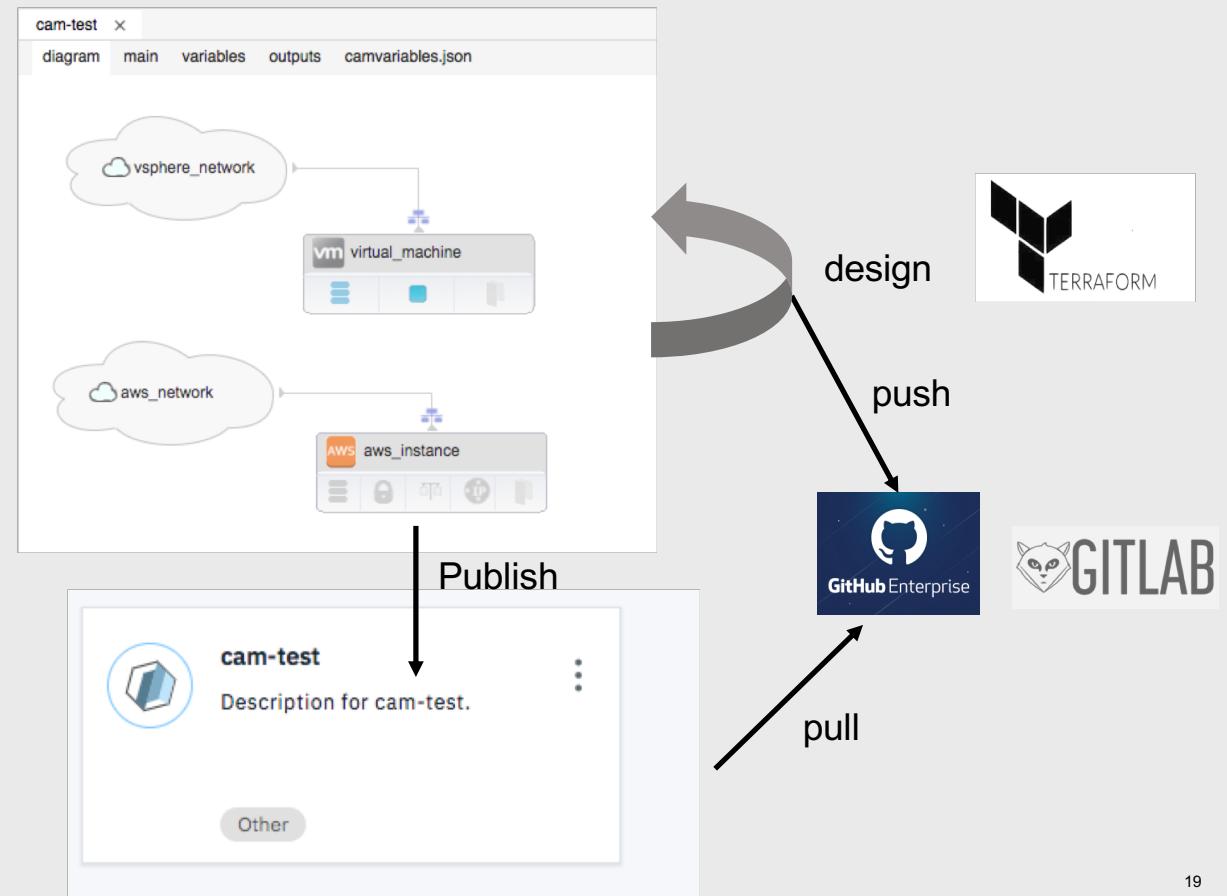
```
cam-infra-template x
diagram main variables outputs camvariables.json
19@ resource "aws_instance" "web-server" {
20  ami = "${var.web-server_ami}"
21  key_name = "${aws_key_pair.auth.id}"
22  instance_type = "${var.web-server_aws_instance_type}"
23  availability_zone = "${var.availability_zone}"
24  subnet_id = "${aws_subnet.subnet.id}"
25  vpc_security_group_ids = ["${aws_security_group.group_name.id}"]
26  tags {
27    Name = "${var.web-server_name}"
28  }
29}
30
31@ resource "aws_vpc" "default" {
32  cidr_block      = "0.0.0.0/0"
33  enable_dns_hostnames = true
34  tags {
35    Name = "${var.network_name_prefix}"
36  }
37}
```



Rich cloud support

# CAM Template Designer – Create CAM Templates

1. Create CAM project
2. Edit Terraform
3. Publish to CAM



# CAM Template Designer – UrbanCode Deploy Integration

## UCD Terraform provisioner

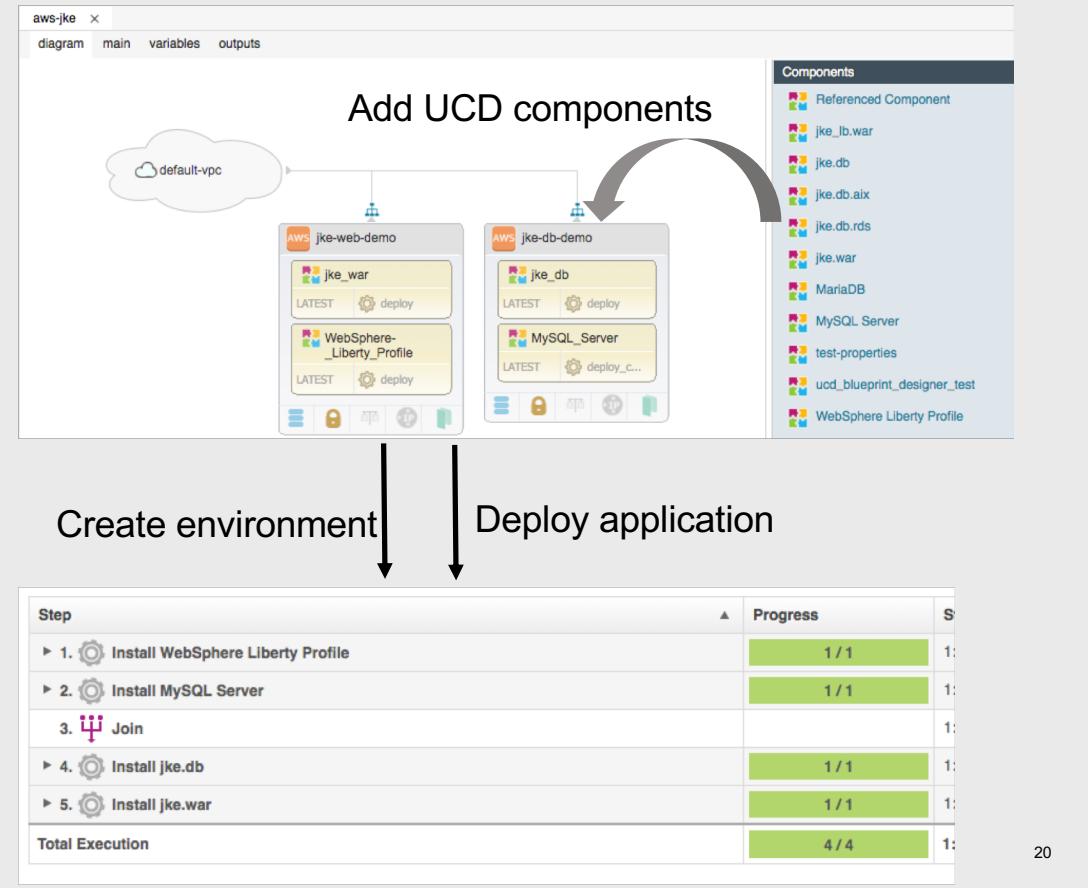
- Install UCD agent
- Included in CAM Terraform runtime

## UCD Terraform provider

- Resources to automate UCD
- Create resource tree, environment, mappings
- Execute application and component processes
- Included in CAM Terraform runtime

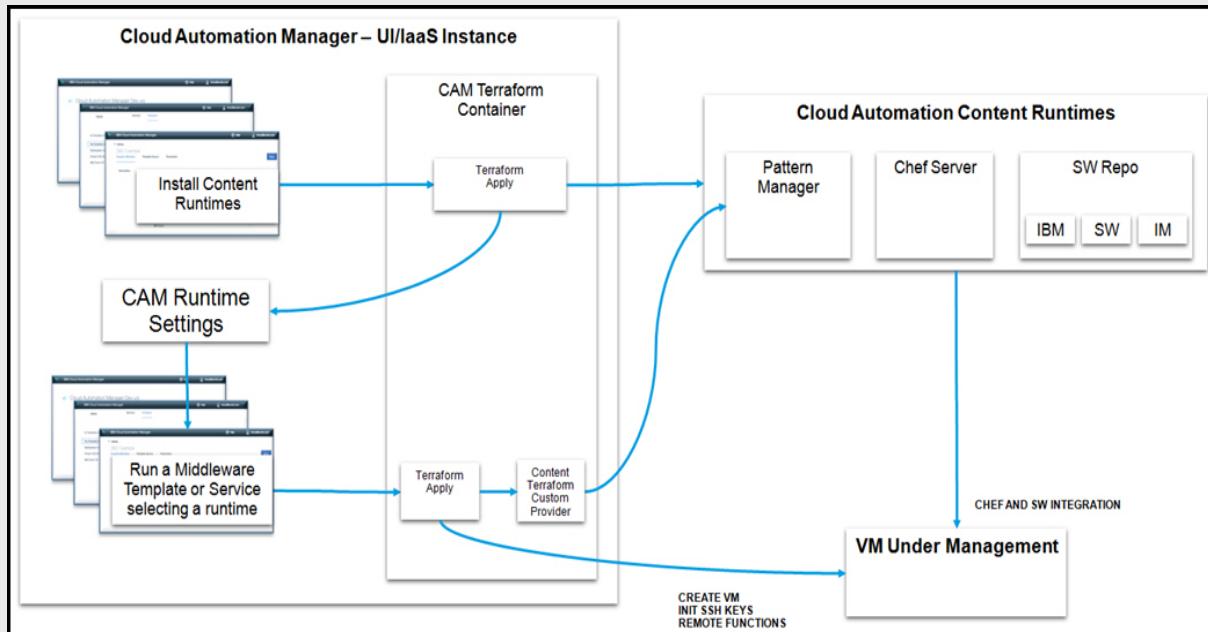
## UCD Designer integration

- View UCD components in palette
- Drag and drop components onto diagram
- Automatically create UCD Terraform resources



# Cloud Automation Manager: Chef content runtime support

## Built-in Chef Server runtime



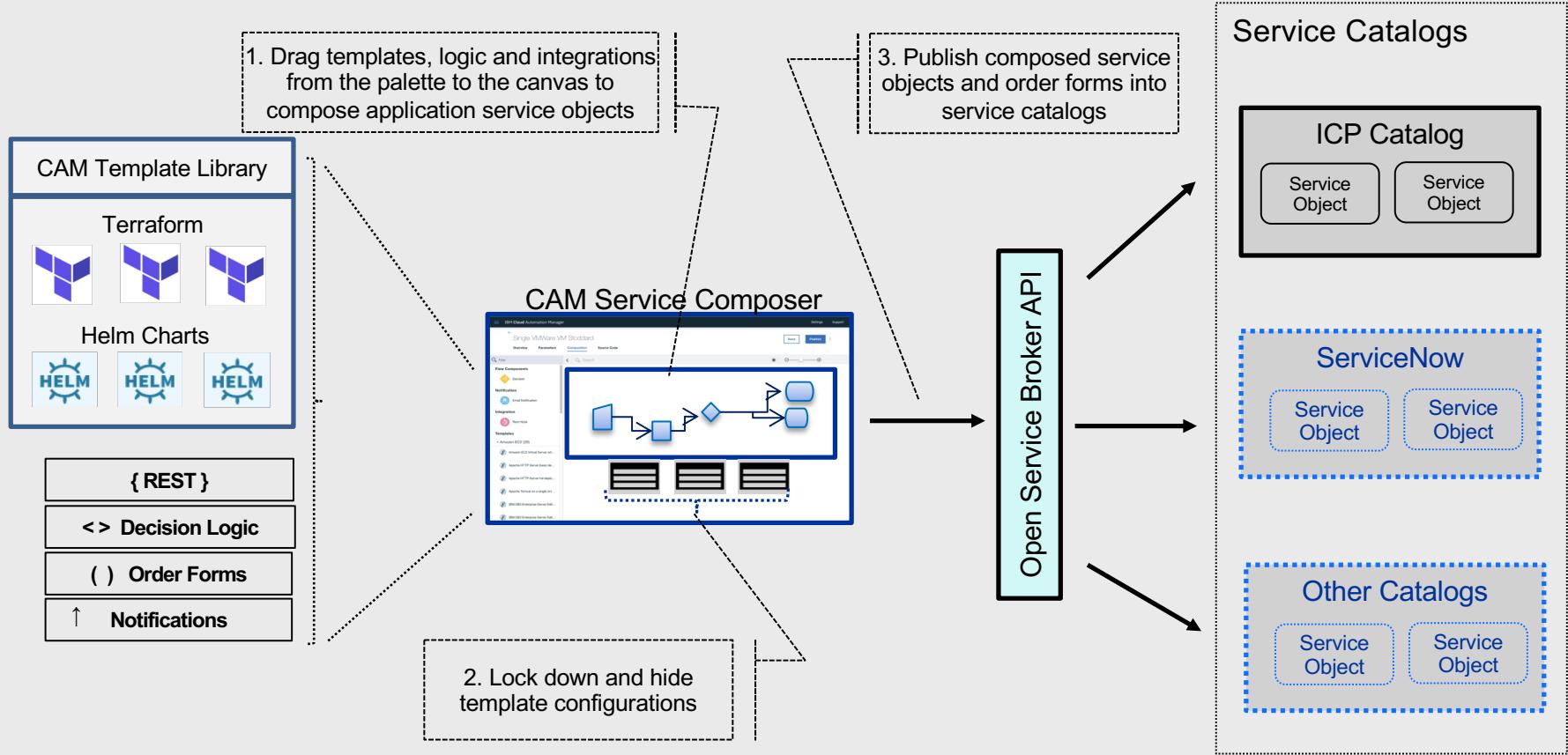
Optional Chef runtime that can be easily deployed into your provider cloud to support IBM Chef enabled content

You supply the virtual server, let CAM stand-up a pre-configure Chef runtime

Fully containerized for maximum portability

Integrated software repository

# Service Composition and Flow Engine



# Assemble, Curate, Publish - with Service Composer

IBM Cloud Automation Manager      Docs      Support      :

WASNDService

Overview    **Composition**    Parameters    Plans & Form    Source Code

Filter    Search

Flow Components

- Decision
- Notification
- Email Notification
- Integration
- Rest Hook
- Templates
- Helm

1. **Assemble** the service by dragging activities from the palette and connecting on the canvas. This service defines three plans.

2. **Curate** the service by locking down configuration variables

3. **Publish** the service into the ICP Catalog

**Development Plan**

**Test Plan**

**Production Plan**

IBM WebSphere Network Deployment V9 on a single ... X

Basic Information    Parameters

Search Parameters

INPUT PARAMETERS	VALUE
WASNode01_dns_servers	1 Items
WASNode01_dns_suffixes	1 Items
WASNode01_domain	\${templates.infoblox35ad464.output.associated_domain}
WASNode01-image	Content/ContentRH_Template_2018_1Q
WASNode01-os_admin_user	root
WASNode01-os_password	Op3nPatterns
user_public_ssh_key	None
WASNode01_root_disk_size	100
WASNode01-name	\${templates.infoblox35ad464.output.associated_}

# Resources

CAM Knowledge Center

<https://www.ibm.com/support/knowledgecenter/SS2L37>

CAM Developer Portal (Blogs, Tutorials, Videos)

<http://developer.ibm.com/cloudautomation>