Cloud Automation Manager

IBM Cloud

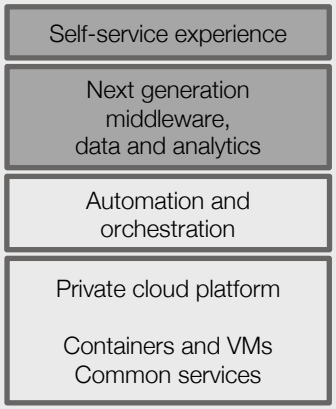
Level Set

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

© 2018 IBM Corporation

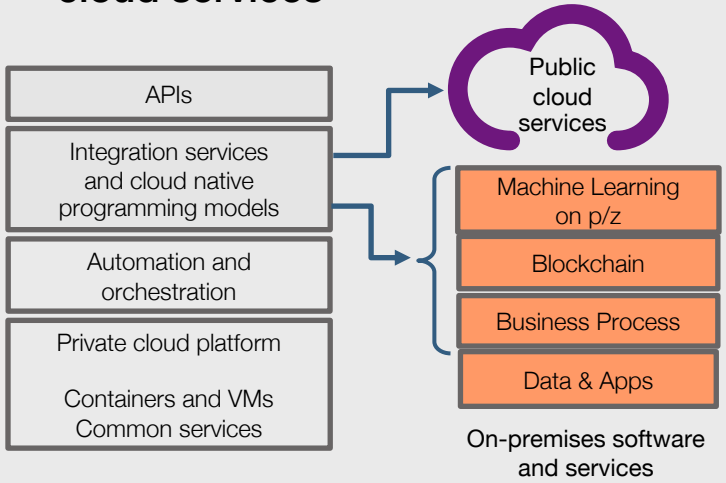
Workload use cases driving private cloud adoption

1. Optimize legacy apps with cloud



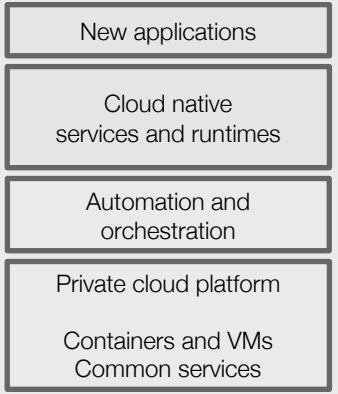
Cloud-enabled middleware

2. Open your datacenter to work with cloud services



Integration and hybrid cloud

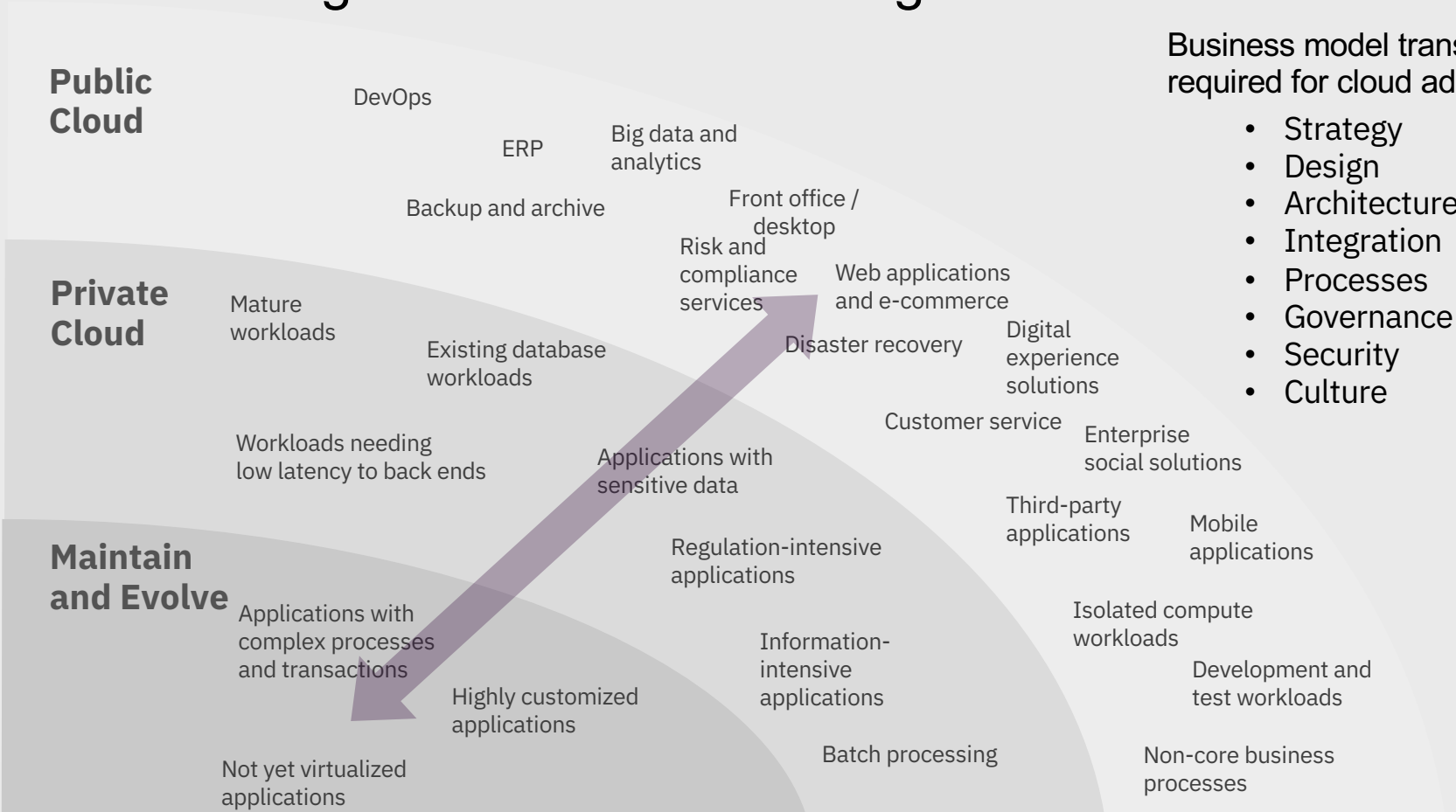
3. Create new cloud native applications



New applications

© 2018 IBM Corporation

Match the right workload with the right cloud



Business model transformation required for cloud adoption:

- Strategy
- Design
- Architecture
- Integration
- Processes
- Governance
- Security
- Culture

* IBM Institute for Business Value study, “Tailoring Hybrid Cloud” August 2016

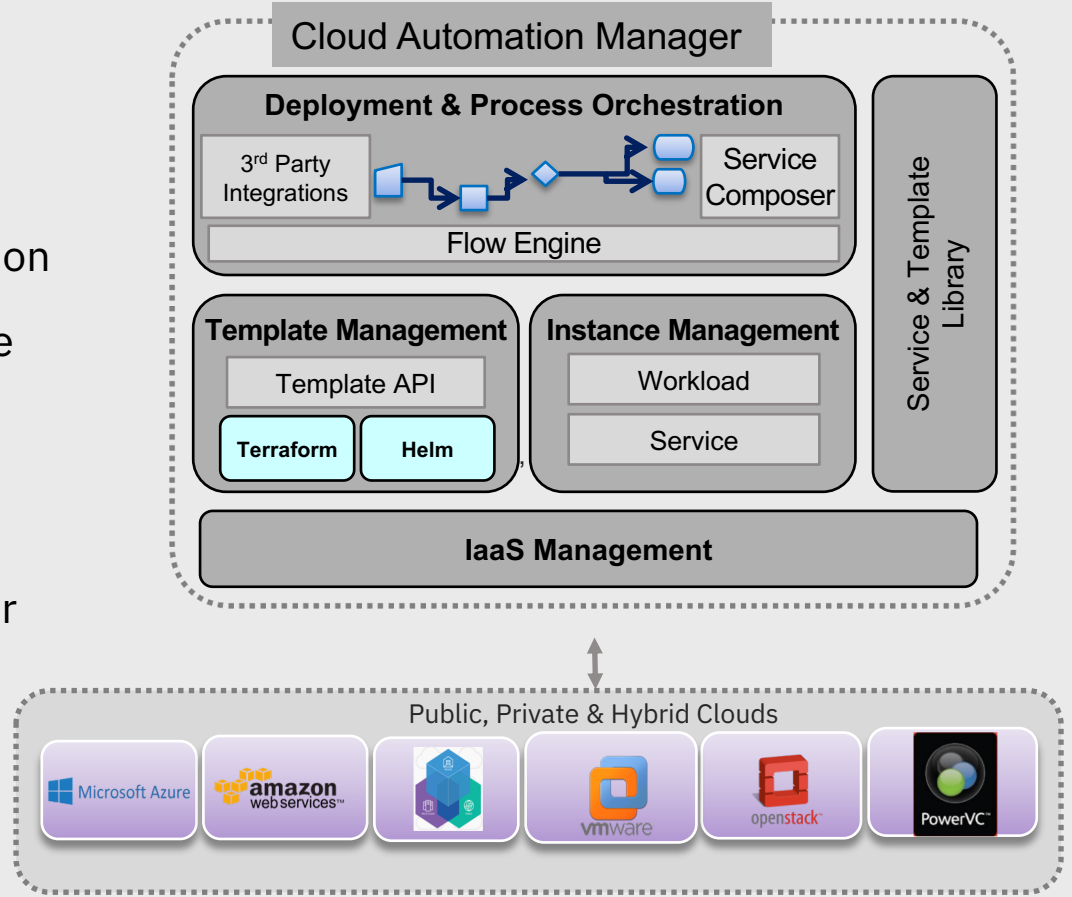
IBM Cloud Automation Manager: Full stack automation and service orchestration

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



© 2018 IBM Corporation

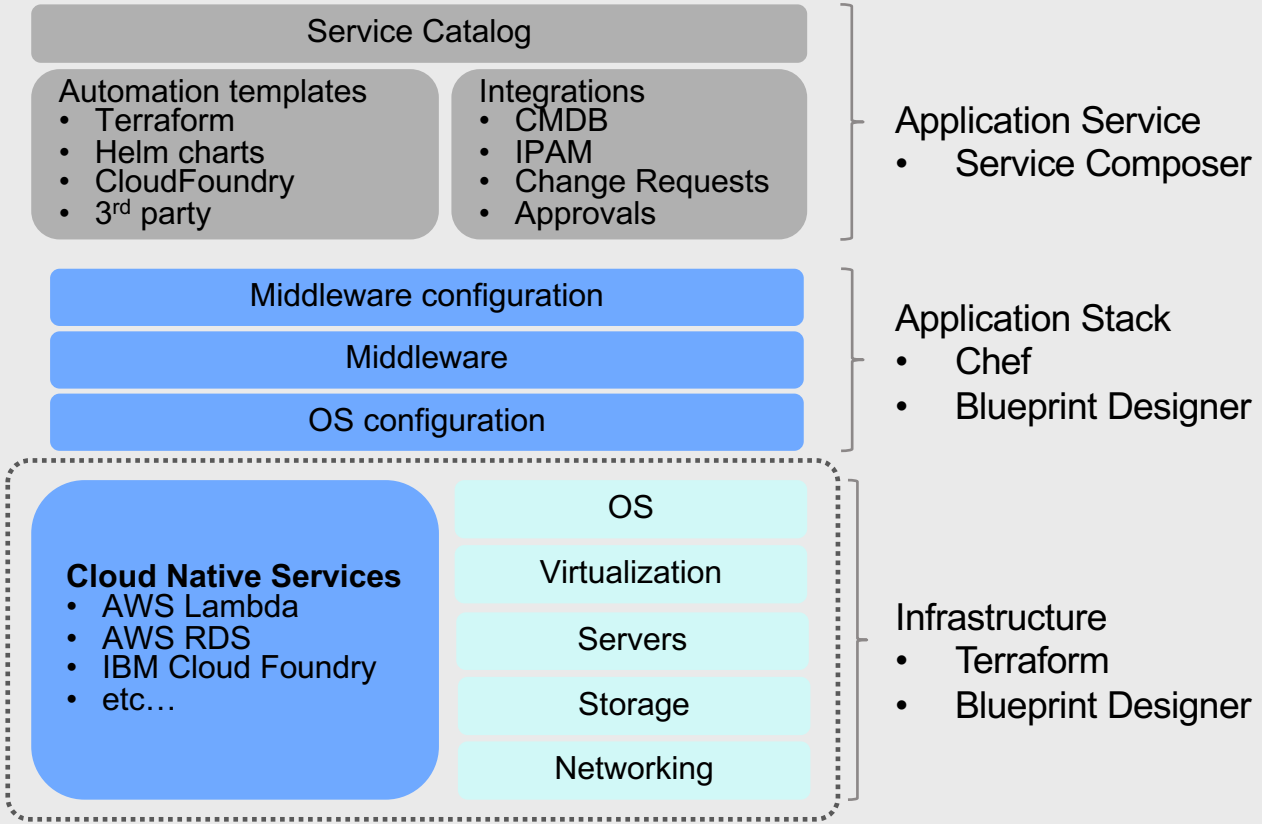
The Cloud Automation Manager business

All clouds, one tool

Any workload architecture

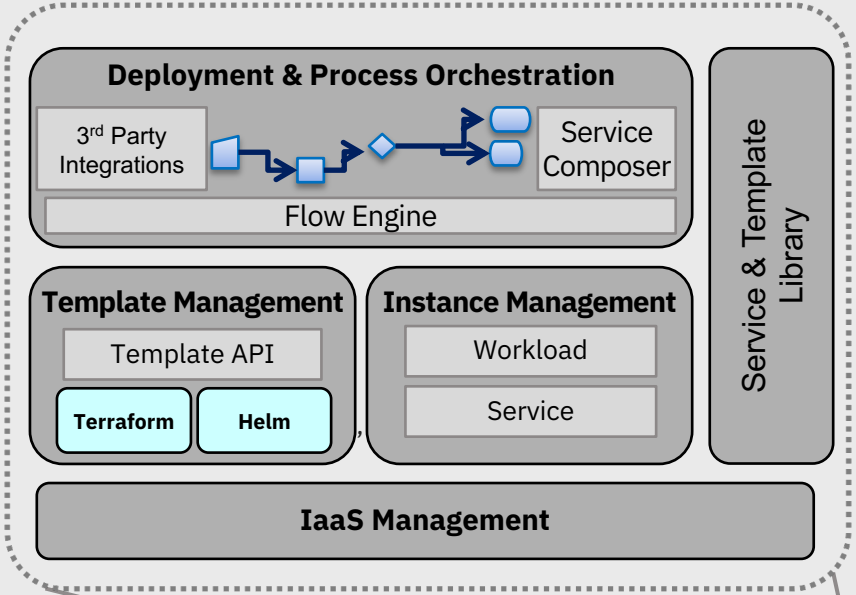
Graphical experience

Purpose build for integration



© 2018 IBM Corporation

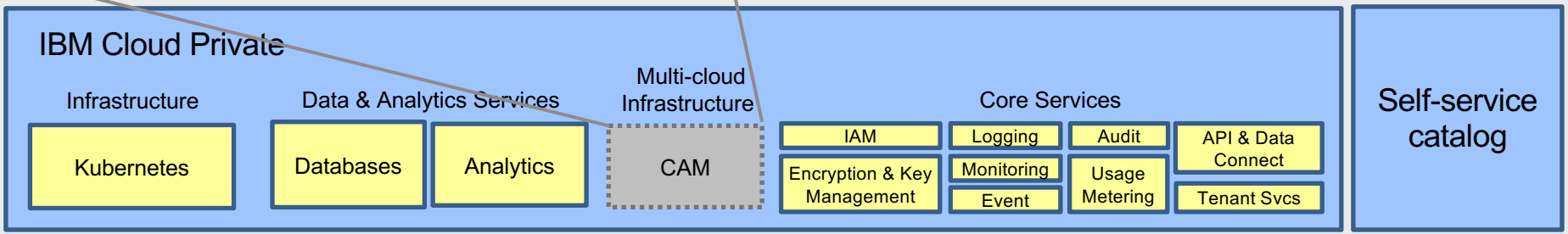
Cloud Automation Manager in IBM Cloud Private



Containerized cloud native application

Installed into an IBM Cloud Private worker node with a Helm chart

Leverages IBM Cloud Private services for enterprise capabilities



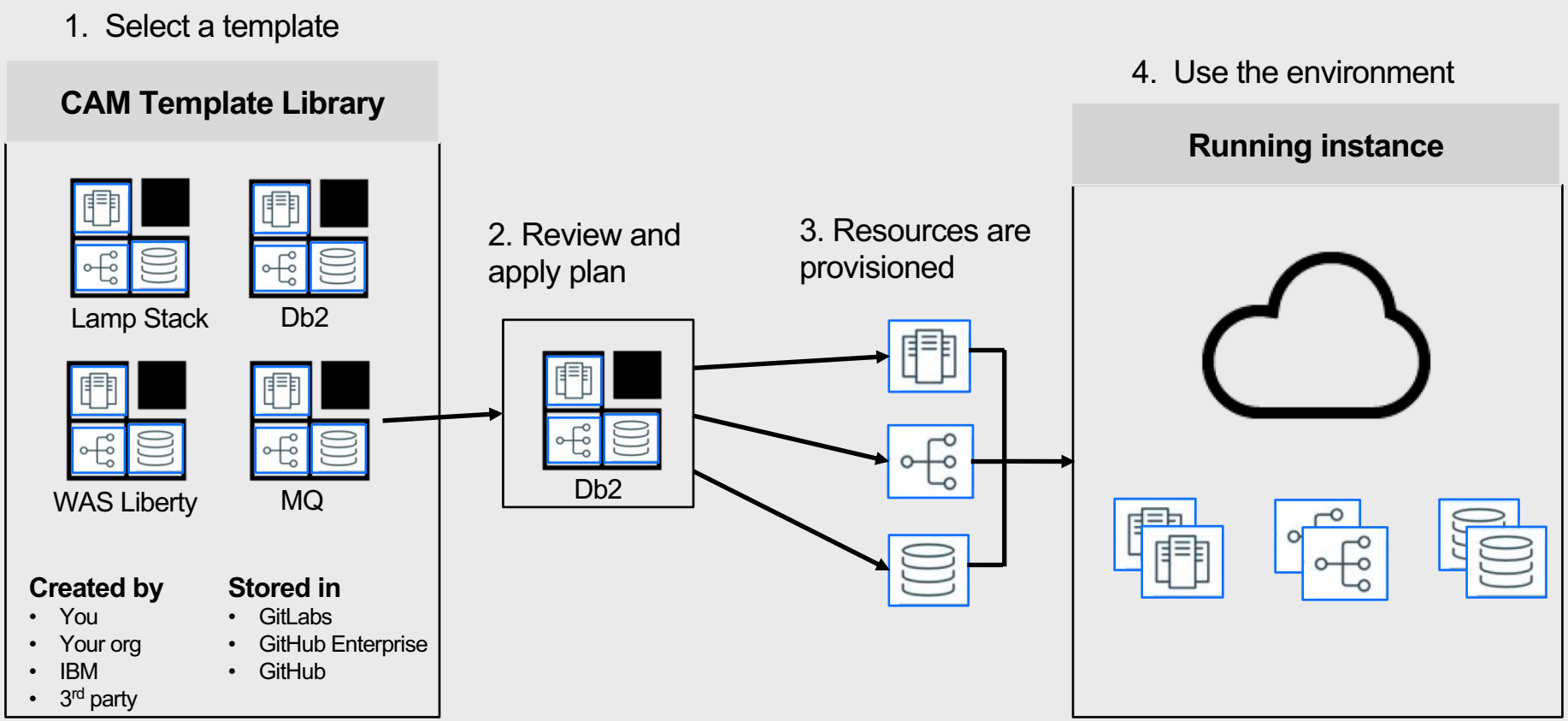
© 2018 IBM Corporation

Automation

The ability to automate the provisioning of resources across multiple clouds, including management of deployments in a single pane of glass, with support of a Software Configuration Manager for software install and configuration.

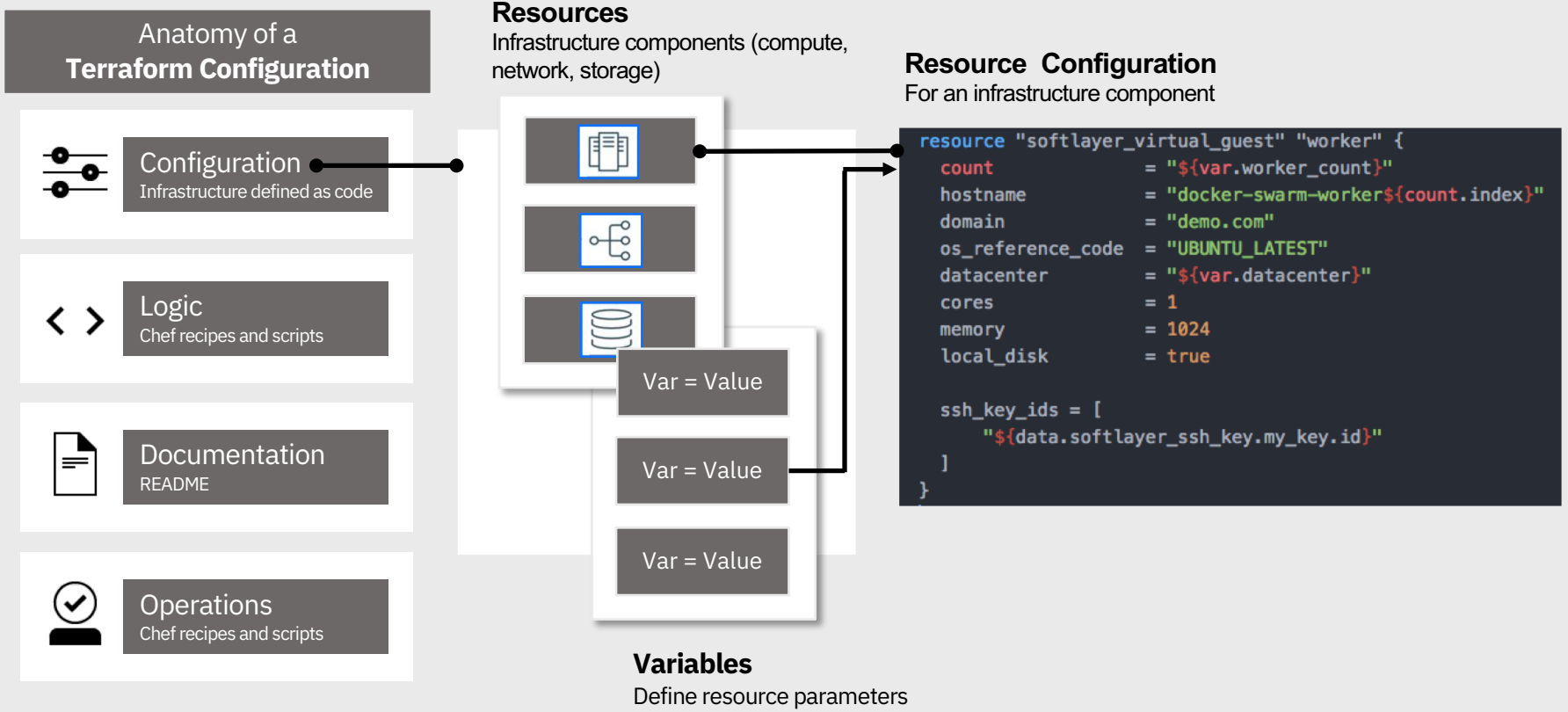
© 2018 IBM Corporation

Using Terraform Configurations



© 2018 IBM Corporation

Anatomy of a Terraform Configuration



© 2018 IBM Corporation

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}"
  ]
}
```

GitHub Enterprise

GITLAB

Store Terraform configuration in Git and manage infrastructure as code

Accelerate development velocity with reusable infrastructure based on open source Terraform

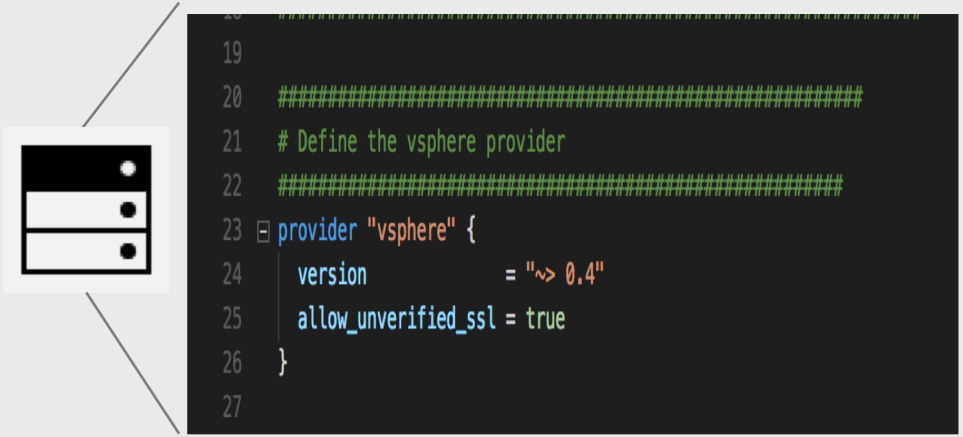
Improve governance and transparency by tracking the ‘who’, ‘what’ and ‘when’ of all environment changes

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

© 2018 IBM Corporation

Managing Terraform Providers



```
19
20  ###############################################
21  # Define the vsphere provider
22  ###############################################
23  provider "vsphere" {
24    version             = "~> 0.4"
25    allow_unverified_ssl = true
26  }
27
```

How to use a specific provider?

1. **Include version attribute on provider section** this will let indicate which version of the provider is use on the template
2. **Provider downloaded from internet** if the version requested is not included in the system.
3. **Allows user to bring newer versions or add new provider** by downloading the provider and loading it into a specific volume used by CAM microservices.

© 2018 IBM Corporation

Shared Data Objects

Administration Objects that allow configuration data from External system that services or templates will interact or need the configuration of that environment

- Example – remote database or LDAP server, Cloud configuration (Vcenter datacenter, storage, network, others)

Template defines the dependency on a data type

- System finds the matching list of objects
- User select one

Template metadata `camvariables.json`

Define dependency on datatype

```
{
    "input_datatypes": [
        {
          "name": "kubeconfig",
          "label": "Kube config"
        }
    ],
```

Define mapping of datatype attribute to variables

```
    },
    {
        "name": "cluster_config",
        "type": "string",
        "description": "Base64 encoded content of the kubeconfig yaml file.",
        "label": "Cluster configuration",
        "secured": true,
        "required": true,
        "hidden": true,
        "default": "${kubeconfig.cluster_config}",
        "immutable": true
```

© 2018 IBM Corporation

Template lifecycle – Plan/Apply

Allows user to make changes on running instances

Two major scenarios:

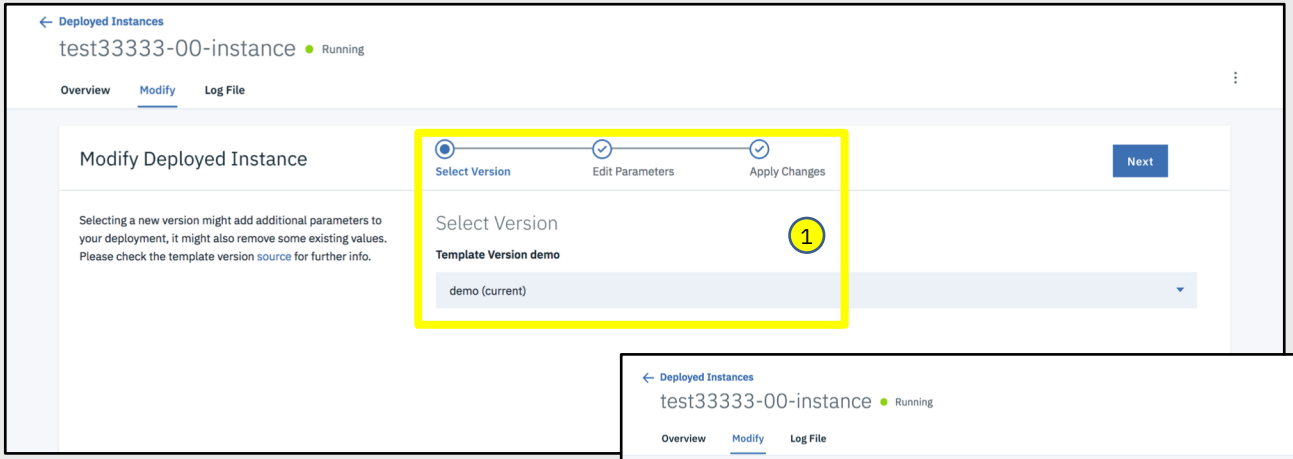
1. User want to change a parameter on an instance
    - Example – change memory/cpu
2. User wants to update to a new version of the template used to create their instance
    - Example – add more VMs or other components

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

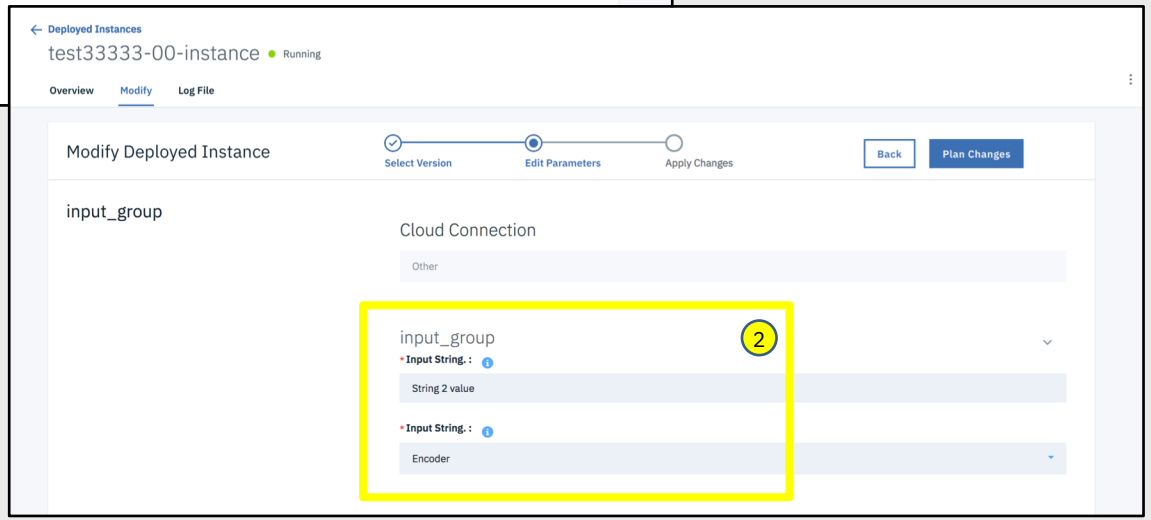
© 2018 IBM Corporation

Plan/Apply



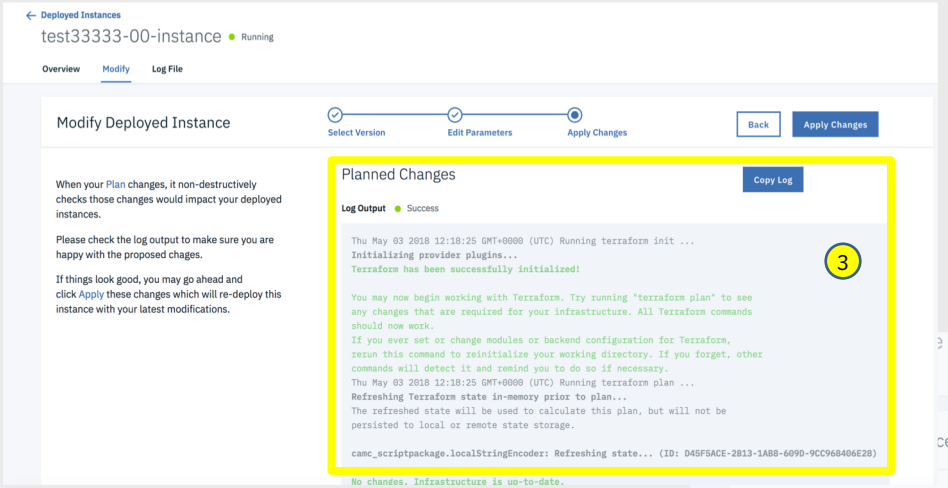
1. Select template version

2. Modify parameters and execute Plan

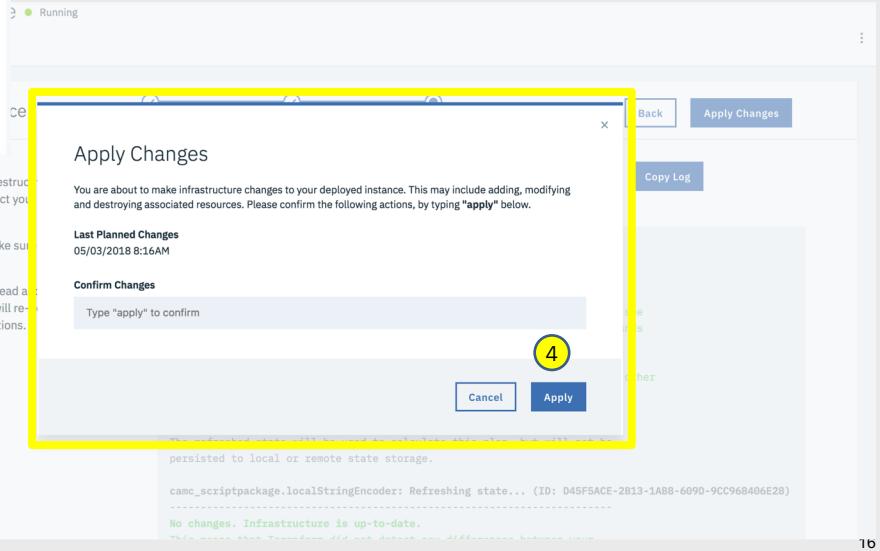


© 2018 IBM Corporation

Plan/Apply (continued)



3. Verify the plan output



4. Apply the changes

© 2018 IBM Corporation

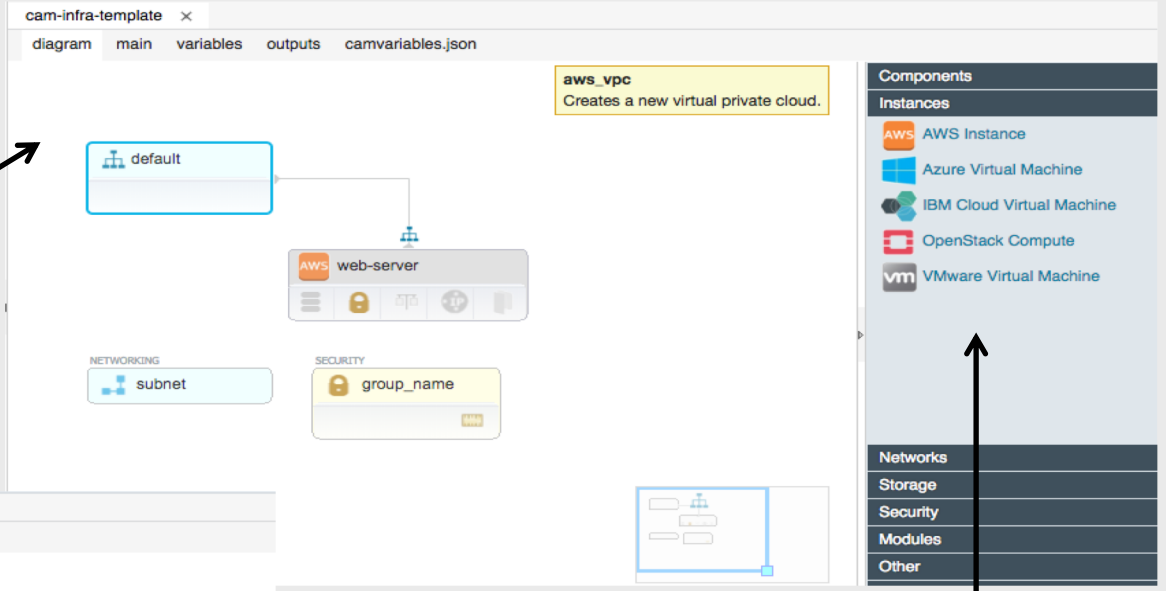
Creating Terraform templates – CAM Template Designer

- 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

© 2018 IBM Corporation

CAM Template Designer – Terraform Design

Graphically create Terraform templates while leveraging Terraform constructs.



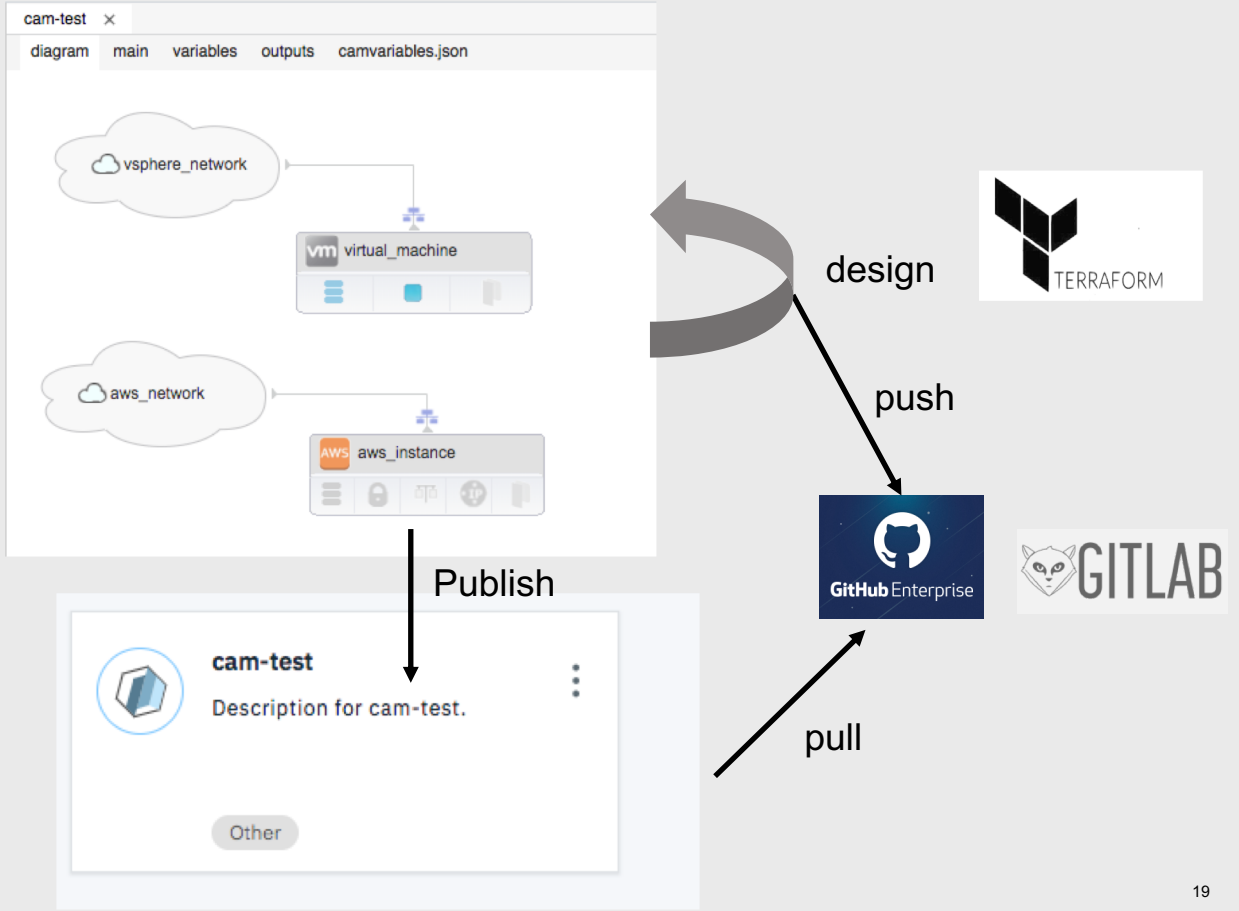

```
resource "aws_instance" "web-server" {
  ami = "${var.web-server_ami}"
  key_name = "${aws_key_pair.auth.id}"
  instance_type = "${var.web-server_aws_instance_type}"
  availability_zone = "${var.availability_zone}"
  subnet_id   = "${aws_subnet.subnet.id}"
  vpc_security_group_ids = ["${aws_security_group.group_name.id}"]
  tags {
    Name = "${var.web-server_name}"
  }
}

resource "aws_vpc" "default" {
  cidr_block           = "0.0.0.0/0"
  enable_dns_hostnames = true
  tags {
    Name = "${var.network_name_prefix}"
  }
}
```

Rich cloud support

© 2018 IBM Cor

CAM Template Designer – Create CAM Templates

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



© 2018 IBM Corporation

CAM Template Designer – UrbanCode Deploy Integration

UCD Terraform provisioner

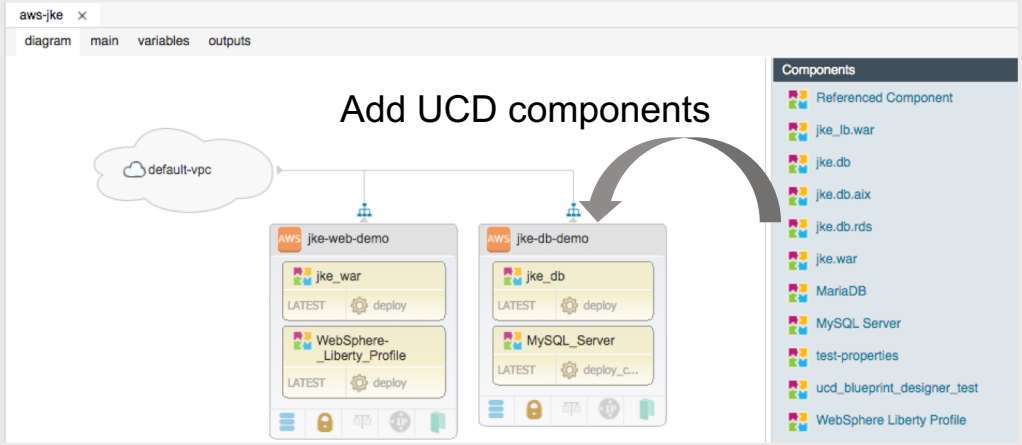
- Install UCD agent
- Included in CAM Terraform runtime

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



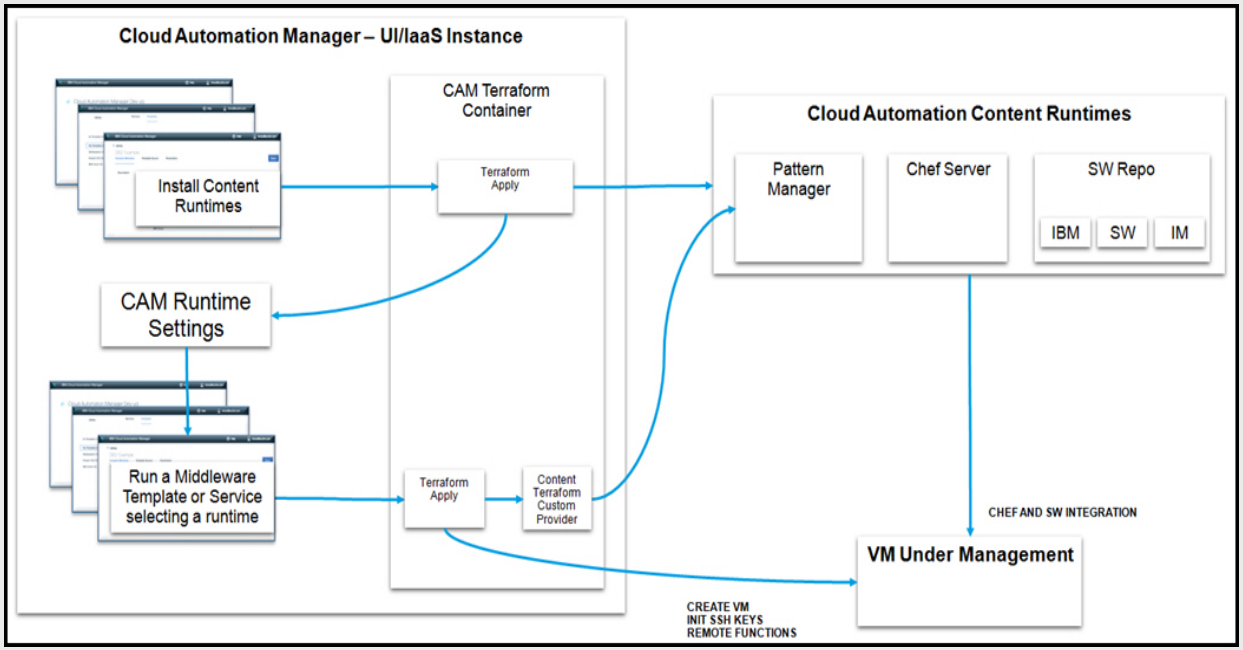
Create environment

Deploy application

© 2018 IBM Corporation

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

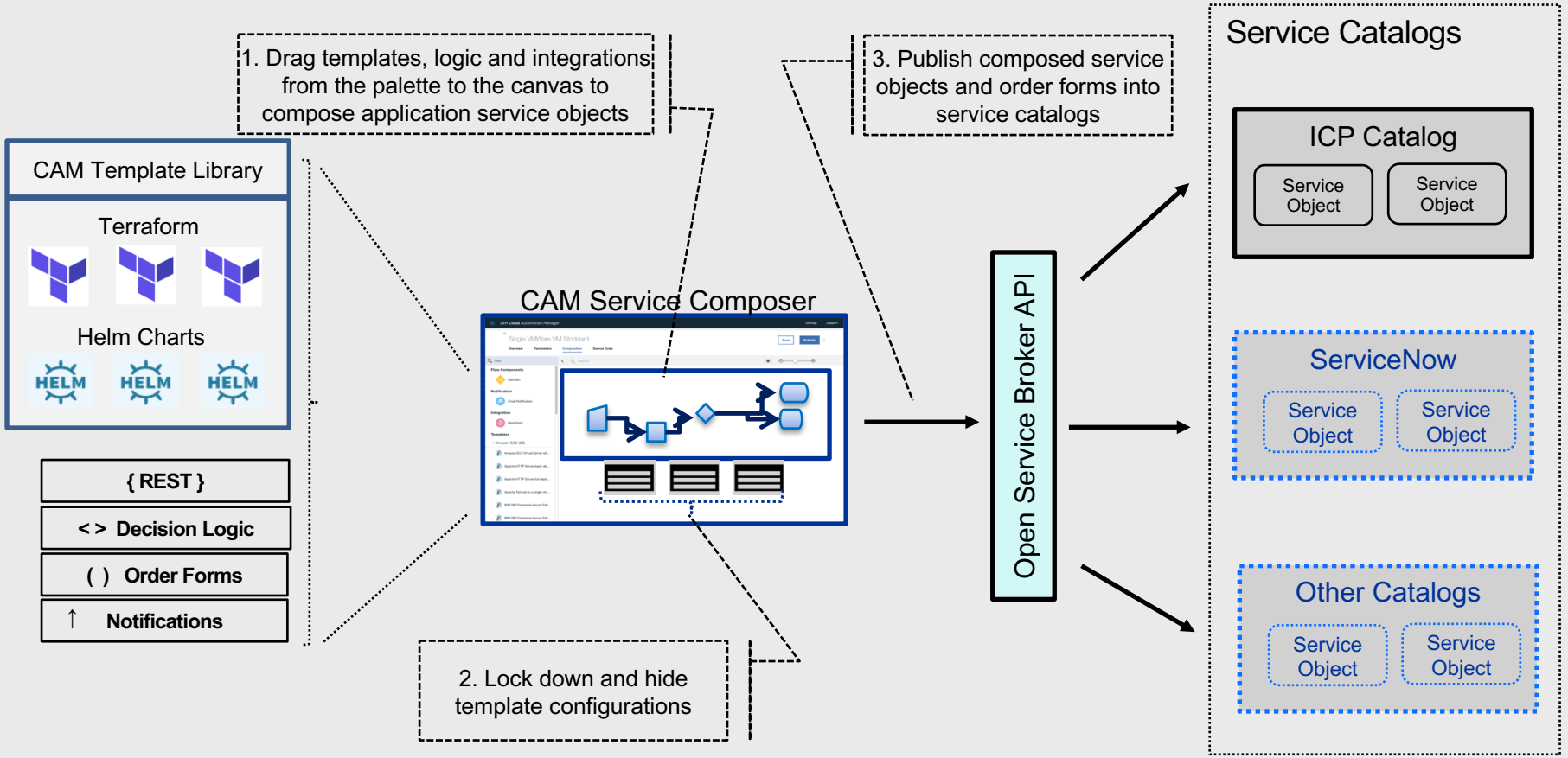
© 2018 IBM Corporation

Orchestration

Assembling automations into a flow that represents the end-to-end lifecycle of a service and supports publication into self service catalogs.

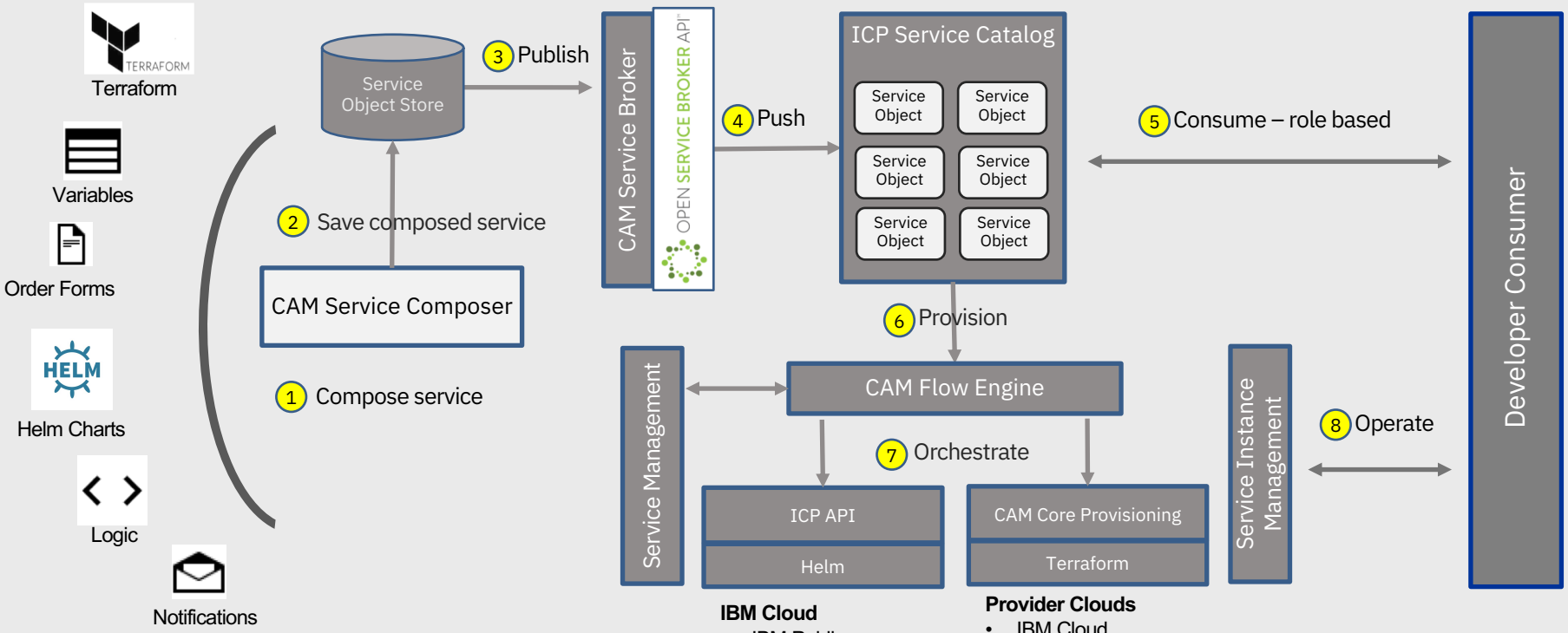
© 2018 IBM Corporation

Service Composition and Flow Engine



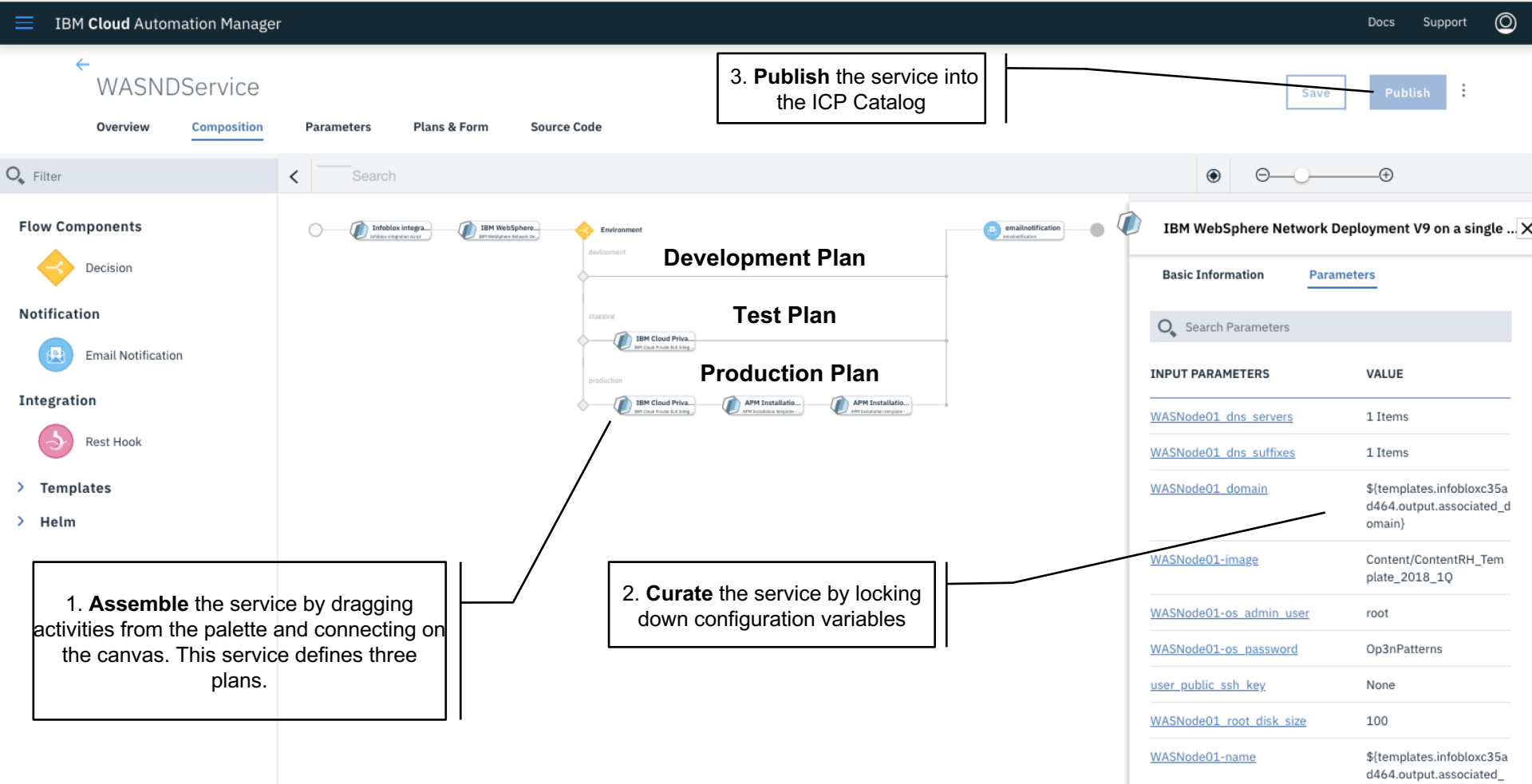
© 2018 IBM Corporation

Service Management Lifecycle: Compose, Publish, Consume and Operate

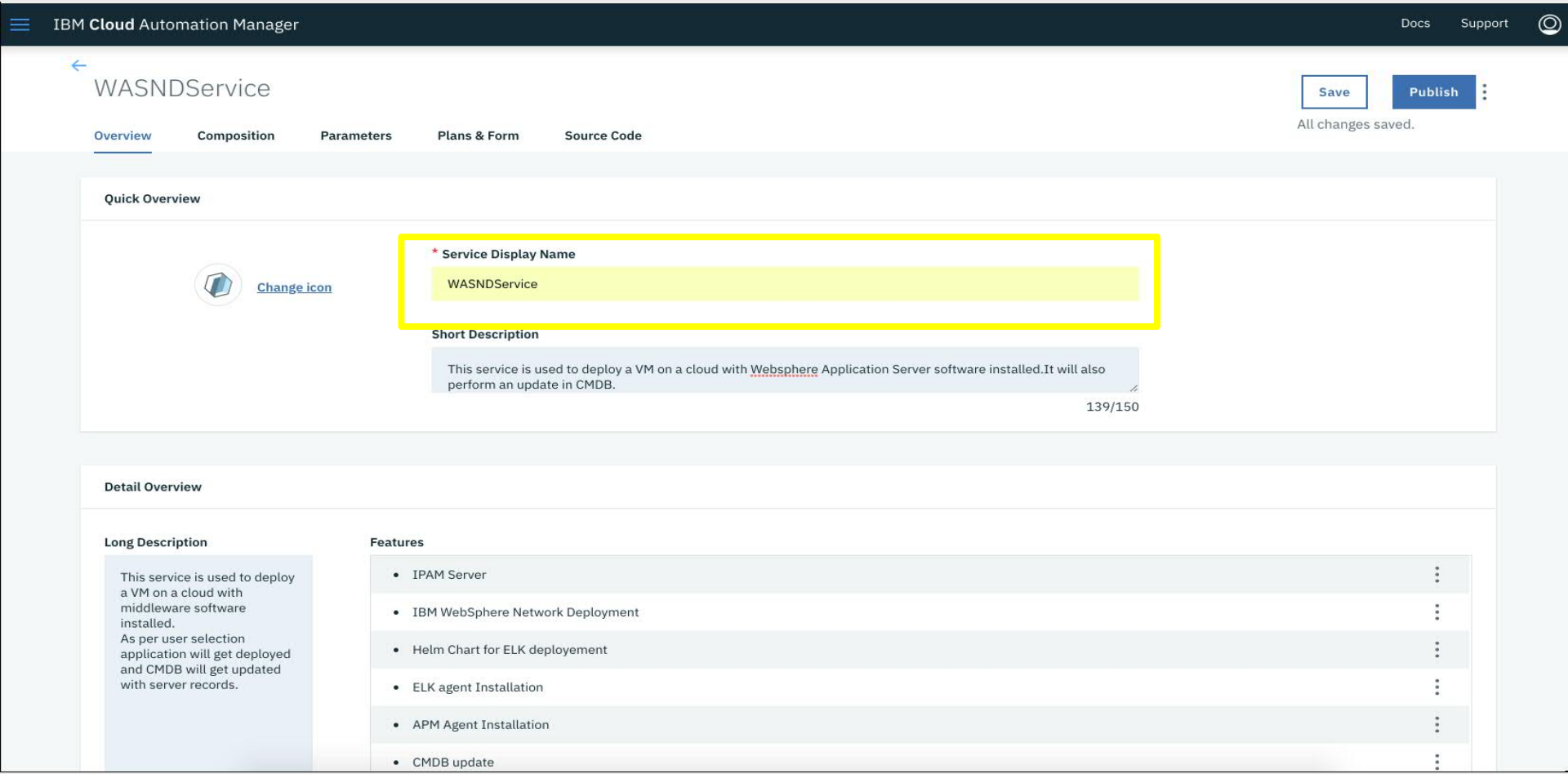


© 2018 IBM Corporation

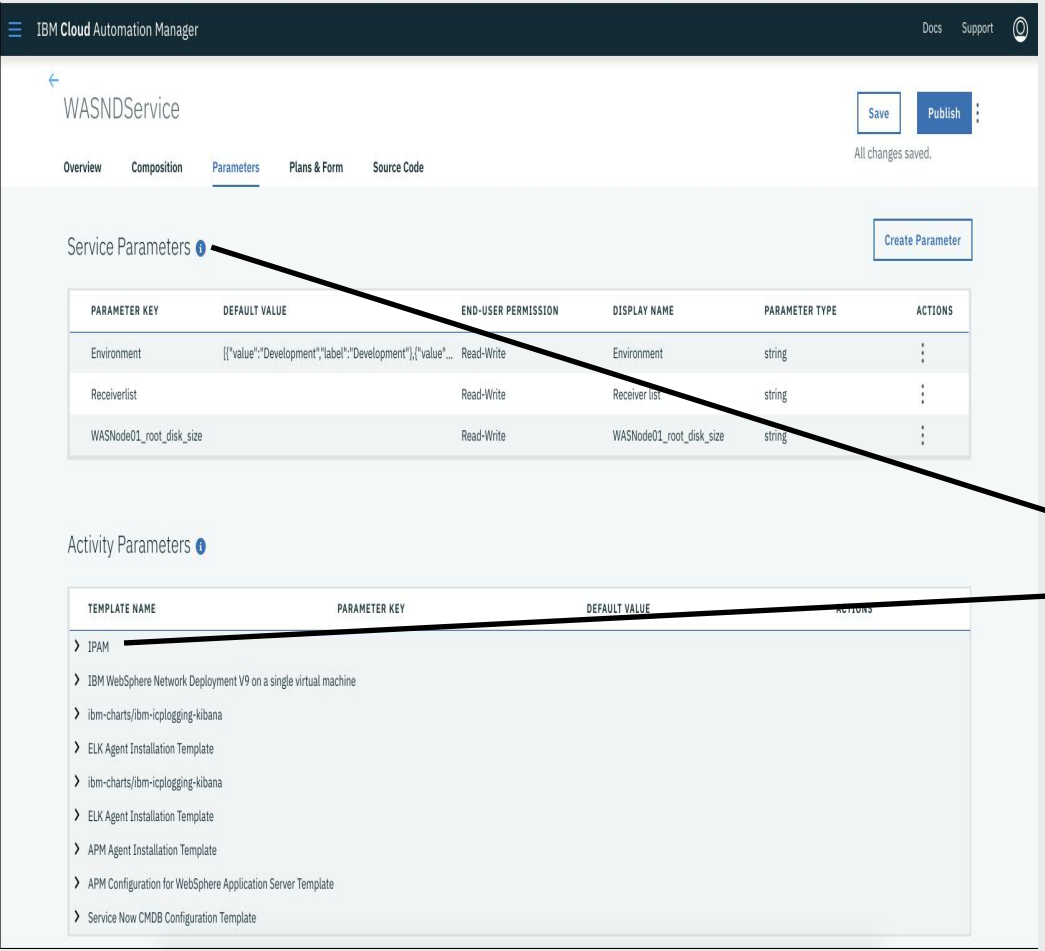
Assemble, Curate, Publish - with Service Composer

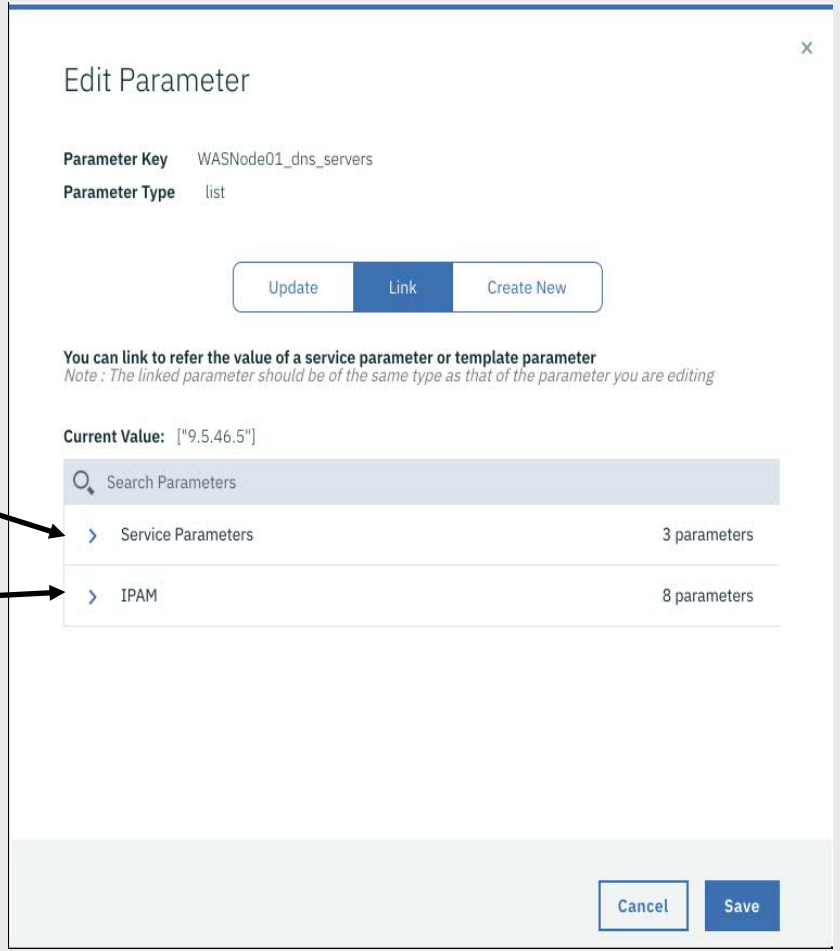


Example: WAS ND Service defining overview for catalog

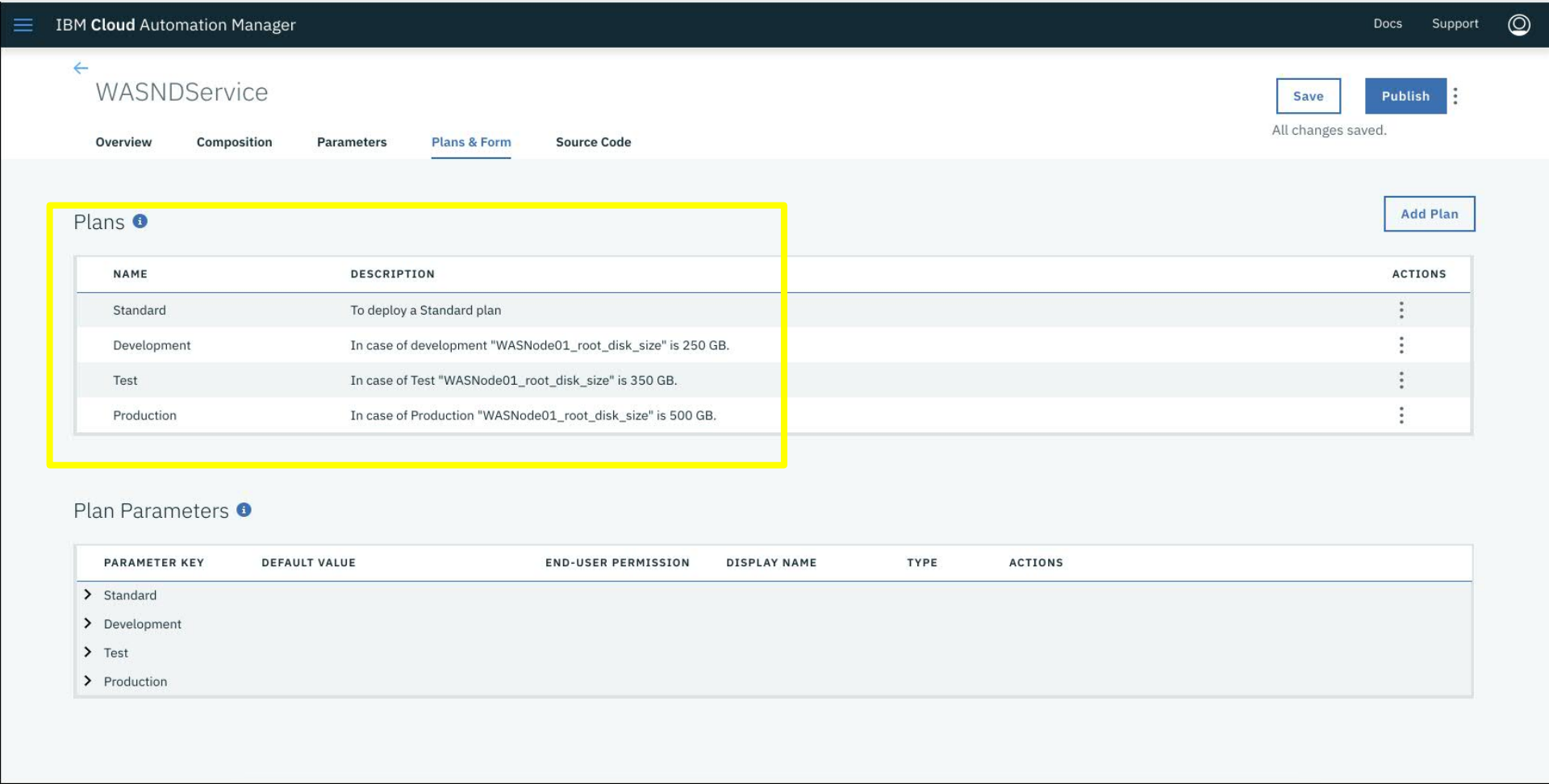


Example: Mapping parameters

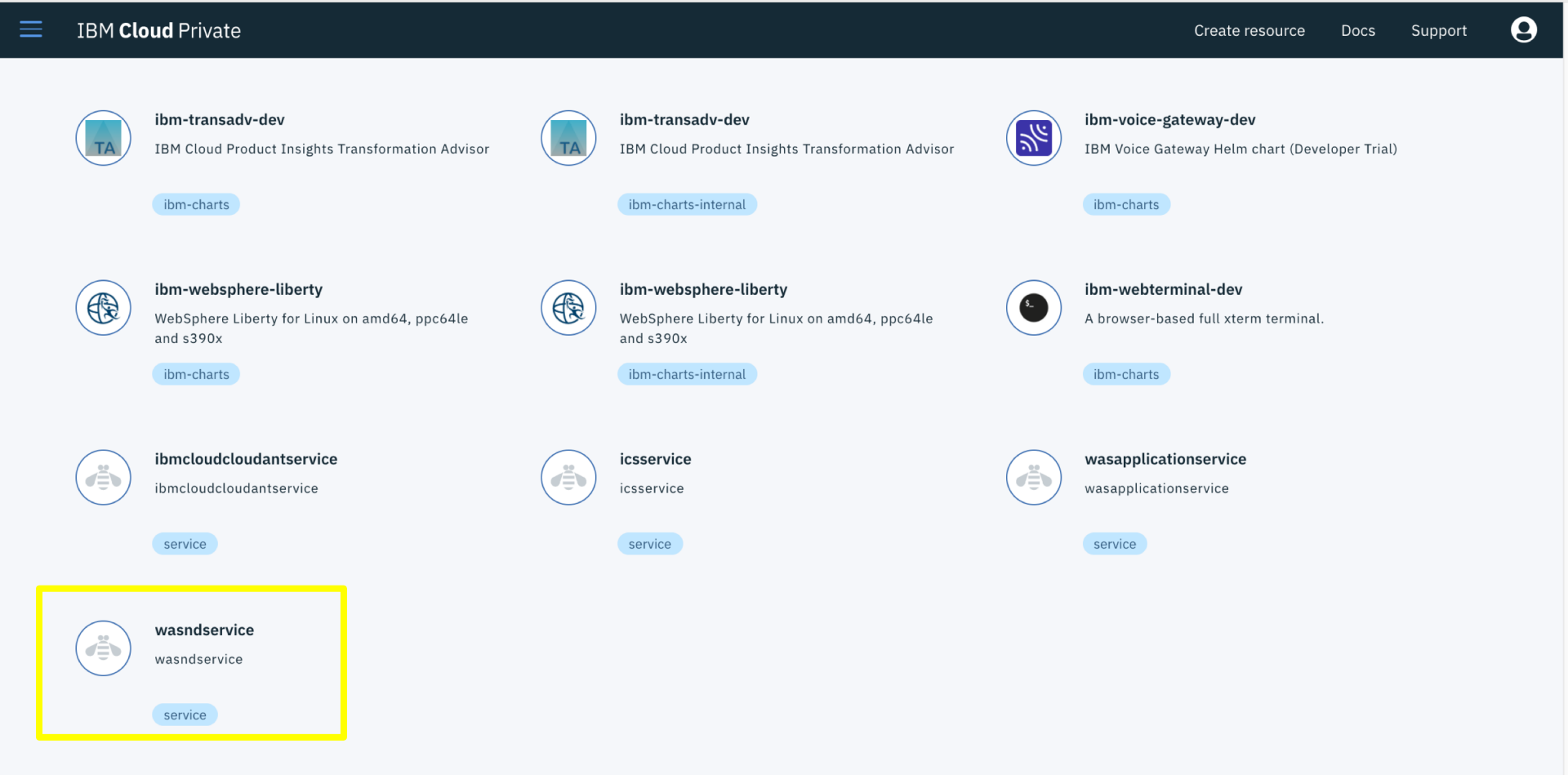




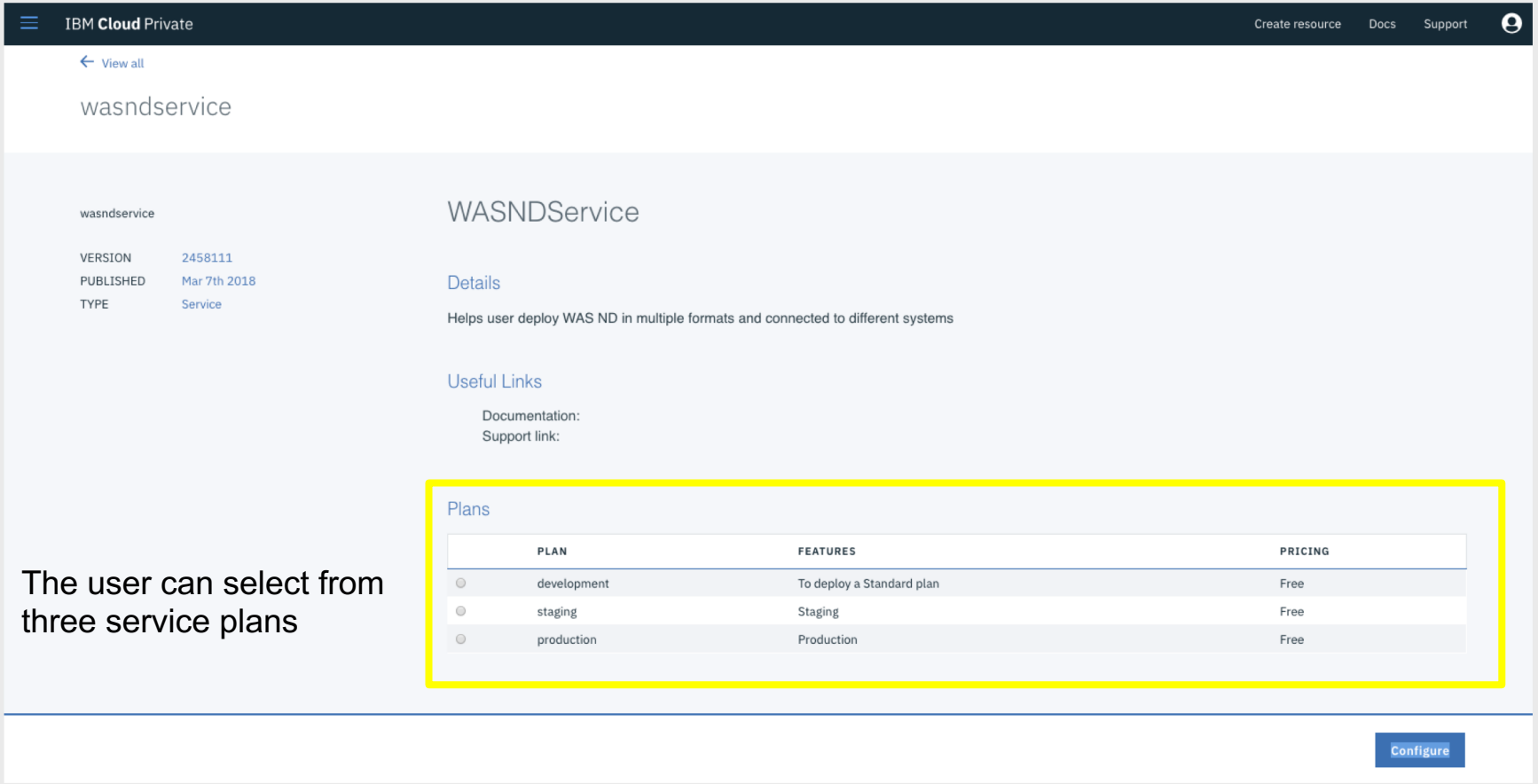
Example: Defining Plans



Example: WAS ND Service in the ICP Service Catalog



Example: Ordering the WAS ND Service from the ICP Service Catalog



The user can select from three service plans

Integration

Integrating Cloud Automation Manager as part of IBM Cloud Private with external components within the datacenter.

© 2018 IBM Corporation

Integration Strategies with Composer

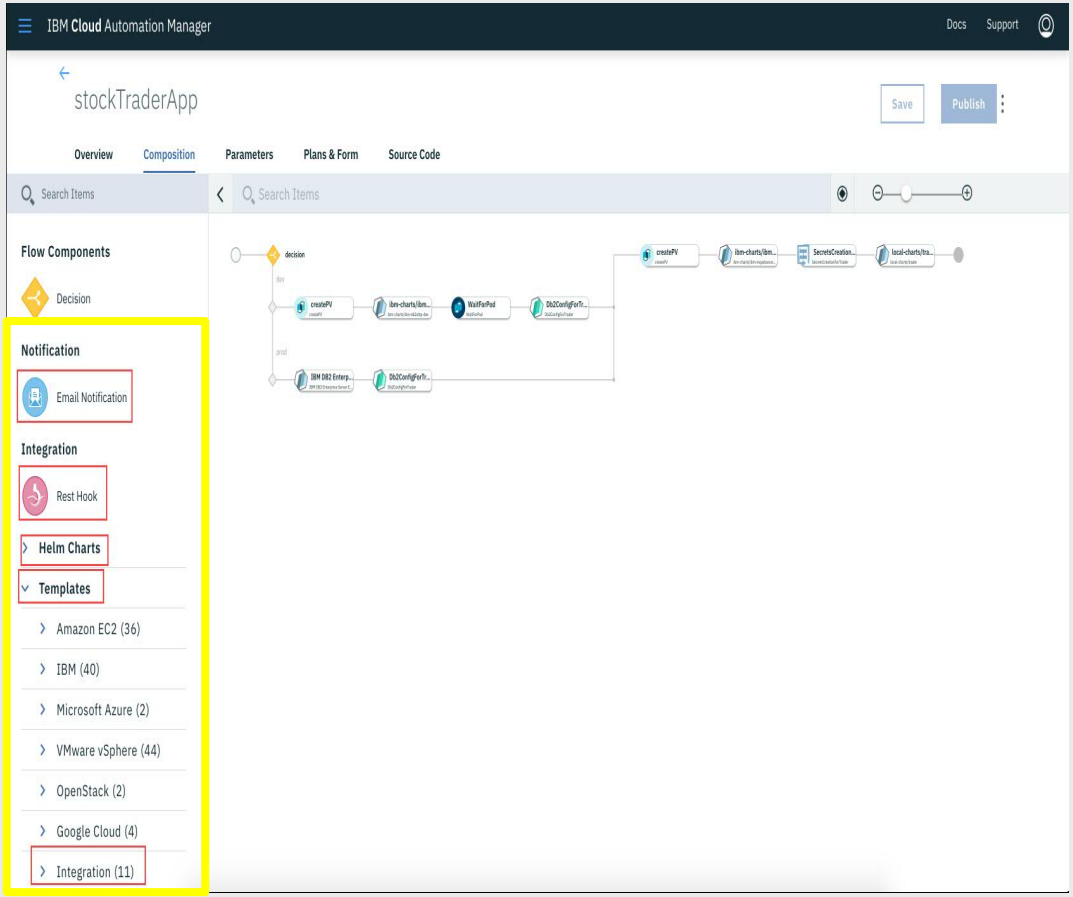
Email notifications to SMTP gateways

REST calls to any external systems

HELM deployment on IBM Cloud Private

Terraform templates on different cloud as IAAS

Terraform templates as scripts on remote / local systems



© 2018 IBM Corporation

CAM Integration via Terraform provider eco-system

Network

Cloudflare
DNS
HTTP
…more

Version Control

Bitbucket
Github
Gitlab
…more

Infrastructure Software

Chef
Consul
Kubernetes
…more

Cloud

CloudStack
Heroku
OpenStack
…more

Community

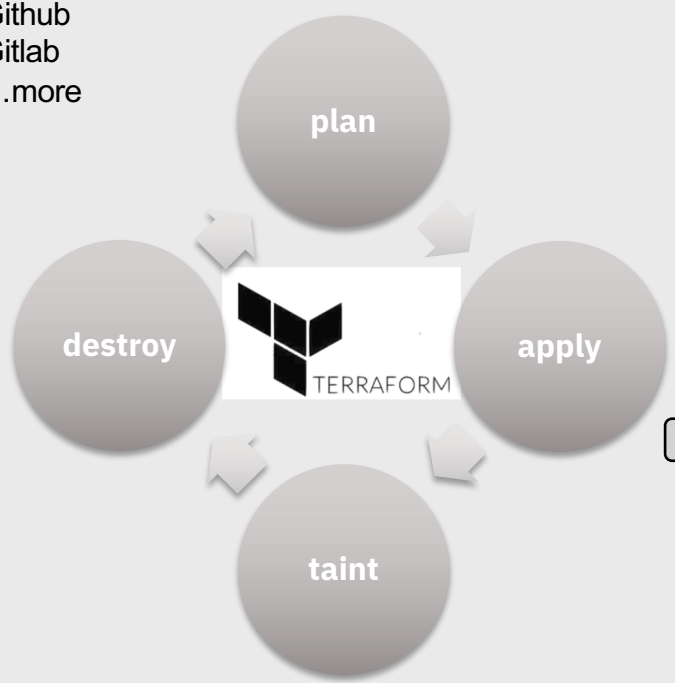
Active Directory
Google Calendar
HP OneView
…more

Major Cloud

AWS
Azure
…more

Monitor and System Mgmt.

Grafana
LogicMonitor
PagerDuty
…more



Misc.

Archive
Cobbler
External
…more

Database

InfluxDB
MySQL
PostgreSQL
…more

© 2018 IBM Corporation

Resources

CAM Knowledge Center

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

CAM Developer Portal (Blogs, Tutorials, Videos)

http://developer.ibm.com/cloudautomation