To the Clouds: Service Orchestration and Cloud Deployment with Juju

Eric Snow ericsnowcurrently@gmail.com

https://bitbucket.org/ericsnowcurrently/presentations/src/default/juju/pycon-2015-poster/
http://goo.gl/sC2TI8

PyCon 2015 Montréal

What is Juju?

Service component deployment, configuration, and coordination, encapsulated in charms. Think of it as “apt-get for the web”.

Why Juju?

Web services can be hard, particularly at scale. Juju provides tools and a language (set of abstractions) that allow you to reason about and manage your infrastructure from a service-oriented perspective, instead of a machine-oriented one.

Rather than reasoning about configuration management, create reference implementations of complex systems that you can discuss in a natural way. Decompose big problems for improved quality and flexibility. Deploy many parts of your stack and let Juju configure (and re-configure) them to talk to each other.

* charm store and ecosystem—a ready-made portfolio of many of the services and solutions you rely on
* public charms published on the store are vetted by Canonical
* CI of published charms provided gratis by Canonical
* only OSI-approved licenses
* “personal namespace” charms also available in charm store
* charms are easily bundled

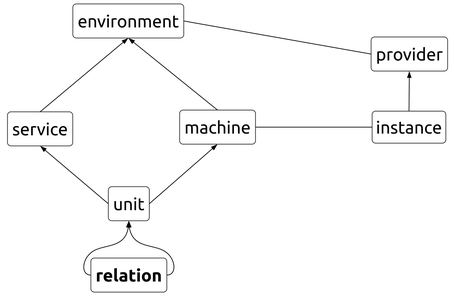
* high-level tools for centralized system orchestration
* powerful GUI (see the demo), which is itself a charm
* easy-to-use CLI

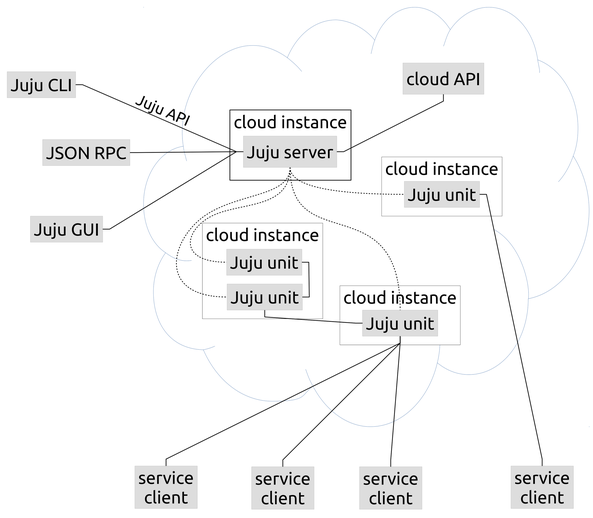
* machine provisioning on popular cloud providers, KVM, containers, and metal
* back-end independence—not tied to a provider or platform
* deploy your full stack locally or to CI with the same configuration as production
* easily move between clouds and infrastructure

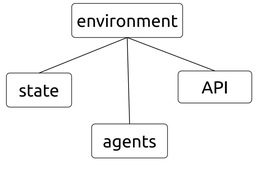
Supported Providers
AWS (EC2, etc.)
OpenStack
Azure
GCE
Joyent
HPCloud
Digital Ocean
VMware (upcoming)
MAAS (bare metal)
local
anything else (“manual”)

Provider Features

Instances
Storage
Networking
Availability Zones
Containers/Virtualization







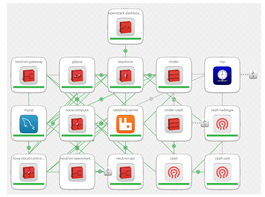
Juju State

* Captures intent for the environment
* Drives events which realize that intent
* Managed by the Juju server (behind API)
* High-availability through replication

Charms

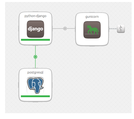
* The key abstraction in Juju
* Models a service using a declarative syntax
* Encapsulates and abstracts away ops knowledge about a service
* Describes how to interact with other services

OpenStack



Big Data

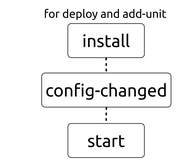
Django



“tmate-slave”

Charm Hooks

Executables called by Juju with access to special env vars and tools.



Python and Juju

* charm helpers (http://pythonhosted.org/charmhelpers/)
* juju quickstart (plugin)
* gui support code (incl. testing)
* deployer (http://pythonhosted.org/juju-deployer/)
* Python API client (http://python-jujuclient.readthedocs.org/en/latest/)
* amulet (https://jujucharms.com/docs/1.20/tools-amulet)
* bundletester (https://github.com/juju-solutions/bundletester)
* CI for charm store
* CI for juju-core
* originally written in Python

Related Solutions and Juju

	salt	ansible	chef
Juju deploys full solution	X	X	X
existing “recipes” charm-able	X	X	X
support in charm-helpers	X	X	

	docker	kubernetes	rails
deploy with Juju	X	X	X
Juju adds coordination	X	X	X
deploy into with Juju	X	X	X

https://fosdem.org/2015/schedule/event/juju_orchestration/
Charles Butler, FOSDEM 2015
video: http://www.youtube.com/watch?v=JOq8YrRUFFc

https://lists.ubuntu.com/archives/juju/2015-March/005132.html
Mark Shuttleworth, [juju-dev] March 2015

https://demo.jujucharms.com/