Deployment Model of Juju

SUNANDA UNNI[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: suunni@indiana.edu*

paper2, March 27, 2017

Developing an application for the cloud is accomplished by relying on the Infrastructure as a Service (IaaS) or the Platform as a Service (PaaS). Juju is a software from Canonical that provides open source service orchestration using a model of IaaS. Juju charms can be deployed for IaaS on cloud services such as Amazon Web Services (AWS), Microsoft Azure and OpenStack. Deep Server provisioning is provided by Juju using MAAS, Metal as a Service. We are exploring the deployment model of Juju.

Keywords: Cloud, I524, Juju, IaaS

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IO-3022/report.pdf

INTRODUCTION

Deploying software component systems [1] is becoming a critical challenge, especially due to the advent of Cloud Computing technologies that make it possible to quickly run complex distributed software systems on-demand on a virtualized infrastructure at a fraction of the cost compared to a few years ago. When the number of software components needed to run the application grows, and their interdependencies become too complex to be manually managed, it is important for the system administrator to use high-level languages for specifying the expected minimal system.

IAAS, PAAS AND MAAS

The IaaS provides a set of low-level resources forming a bare computing environment. Developers pack the whole software stack into virtual machines containing the application and its dependencies and run them on physical machines of the provider's cloud. Exploiting the IaaS directly allows a great flexibility but requires also a great expertise and knowledge of the cloud and application entities involved in the process IaaS describes the provision of processing, storage and networking (and potentially) other basic computing resources, over a network and in an on-demand fashion. An example of IaaS is the AWS[2], Juju.

IaaS customers does not host or manage the dedicated or virtual server. Figure 1 for Juju scheme of work as IaaS.

In PaaS (e.g., Google App Engine [3], Azure [4]) a full development environment is provided. Applications are directly written in a programming language supported by the framework offered by the provider, and then automatically deployed to the cloud. The high-level of automation comes however at the price of flexibility: the choice of the programming language to use is restricted to the ones supported by the PaaS provider,

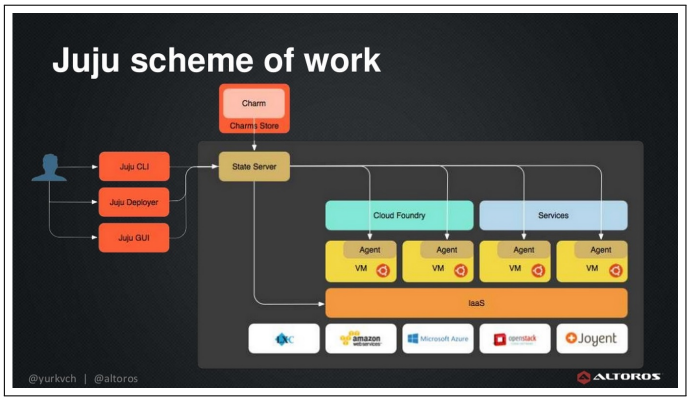

Fig. 1. Basic Deployment of Juju scheme of work.

and the application code must conform to specific APIs.

JUJU

In the IaaS, two deployment approaches [1] are gaining more and more momentum: the holistic and the DevOps one. In the former, also known as model-driven approach, one derives a complete model for the entire application and the deployment plan is then derived in a top-down manner. In the latter, put forward by the DevOps community [5], an application is deployed by assembling available components that serve as the basic building blocks. This emerging approach works in a bottom-up direction: from individual component descriptions and recipes for installing them, an application is built as a composition of these recipes.

One of the representative for the DevOps approach is Juju [6], by Canonical. It is based on the concept of charm: the atomic

Fig. 2. Juju installation steps[6]

```
$ sudo apt-get update && sudo apt-get install juju
```

Fig. 3. Juju bootstrap command with yaml configuration set [6].

```
$ sudo juju generate-config
$ edit the environments.yaml file
$ sudo juju sync-tools
$ sudo juju bootstrap
```

unit containing a description of a component.

In 2.0, Juju follows a Controller-Model type of configuration. A Juju controller is the management node of a Juju cloud environment. In particluar, it houses the database and keeps track of all the models in that environment. Although it is a special node, it is a machine that gets created by Juju (during the "bootstrap" stage) and, in that sense, is similar to other Juju machines.

A Juju model is an environment associated with a controller. During controller creation two models are also created, the 'controller' model and the 'default' model. The primary purpose of the 'controller' model is to run and manage the Juju API server and the underlying database. Additional models may be created by the user.

Since a controller can host multiple models, the destruction of a controller must be done with ample consideration since all its models will be destroyed along with it.

JUJU INSTALLATION

Installing Juju is straight forward, it is described in detail in Getting Started document [7] and Paper regarding deploying Juju on MaaS [8].

The command to install Juju is shown in Fig 2.

Not covering setup of LXD or MaaS for the physical containers or cluster on which Juju is to be installed. We can refer to [7] for LXD and [8] for MaaS installation.

To create a controller, juju bootstrap command is used. Two ways to bootstrap

1. Generate config with the environment specified in yaml, as shown in Fig 3.
2. By specifying the details on command line as shown in Fig 4.

INSTALLATION OF CHARMS

Fig 5 shows the deployment model of Charms.

When deploying charms you can do the following-

Fig. 4. Juju bootstrap command with command line settings [6].

```
$ sudo juju bootstrap localhost lxd-test

lxd-test is the controller on the local machine.
```

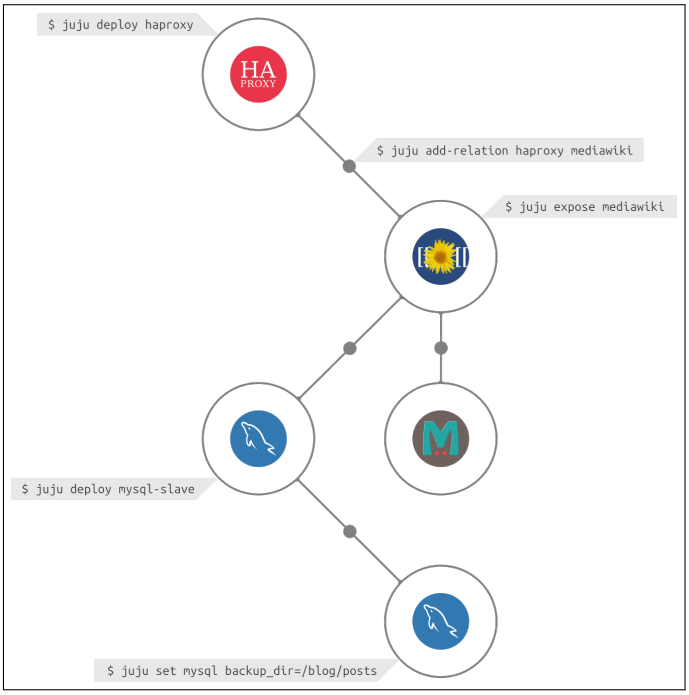

Fig. 5. Charms and Bundles [6].

Fig. 6. Juju Charm deploy from external repository [6].

```
$ sudo apt-get install charm-tools
$ charm get <charm>
$ juju deploy local:trusty/<charm-name>
```

1. Download them from external repositories as you deploy. For direct installation from external repository, we can use Fig 6.
2. Download them first to a local repository and then point to the repository as you deploy. For deploying from local repository you can install bazar as shown in Fig 7.

Status of installation can be checked by command as shown in Fig 8.

On successful installation of MySQL and mediawiki, we can see the status as shown in Fig 9.

BUNDLING

A Bundle is an encapsulation of complex deployments including many different applications and connections. Bundles are deployed as-

1. From the Juju Charm Store as shown in Fig 10.

Fig. 7. Juju Charm deploy from local repository [8].

```
$ sudo apt-get install bzr
$ mkdir -p /opt/charms/trusty; cd /opt/charms/trusty
$ bzr branch lp:charms/<charm-name>
$ juju deploy repository=/opt/charms local:trusty/<charm-nar
```

Fig. 8. Juju Charm installation status [6].

```
$ juju status
```

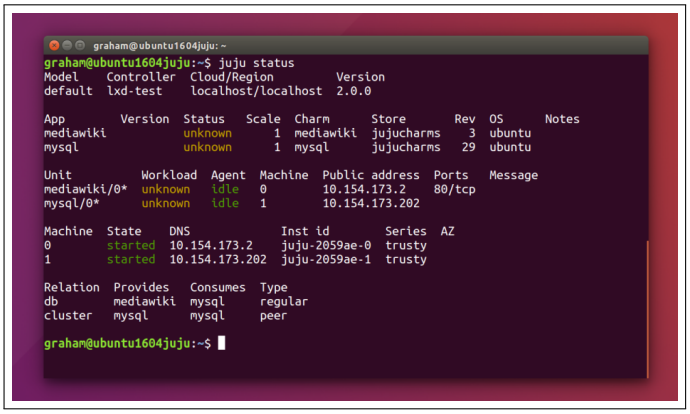

Fig. 9. Juju status post installation of Bundle [7].

2. By exporting the current configuration into a bundle. Juju-UI can be used for exporting the current configuration

ADVANTAGES OF JUJU

Scaling the setup is possible. No prior knowledge of application stack required. Charm store provides charms for a lot of common Opensource applications. Bundling helps in easy re-deployment to new environment. Juju works with the exiting Configuration management tool.

DRAWBACKS OF JUJU

In order to use Juju, some advanced knowledge of the application to install is mandatory. This is due to the fact that the metadata does not specify the required functionalities needed by a component. Currently Juju can be used for deployments on limited OS - Windows, CentOS, and support for Ubuntu. Juju does not still support handling of circular dependencies

CONCLUSION

For Application provisioning Juju does provides orchestration similar to Puppet, Chef, Salt etc. Advantage is Juju comes with GUI in open source version and is interoperable with most of the major cloud providers.

REFERENCES

[1] T. A. Lascu, J. Mauro, and G. Zavattaro, "Automatic deployment of component-based applications," *Science of Computer Programming*, vol. 113, pp. 261–284, 2015.

[2] Amazon Web Services, Inc., "Amazon web services," Web Page, 2017. [Online]. Available: https://aws.amazon.com/

[3] Google Developers, "Google app engine documentation," Web Page. [Online]. Available: https://developers.google.com/appengine

[4] Microsoft, "Microsoft azure," Web Page, 2017. [Online]. Available: http://azure.microsoft.com

[5] Mediaops, LLC, "Devops- where the world meets devops," Web Page, 2017. [Online]. Available: https://devops.com

[6] Canonical Ltd, "Ubuntu cloud documentation," Web Page. [Online]. Available: https://www.ubuntu.com/cloud/juju

[7] ——, "Getting started with juju," Web Page. [Online]. Available: https://jujucharms.com/docs/stable/getting-started

[8] K. Baxley, J. la Rosa, and M. Wenning, "Deploying workloads with juju and maas in ubuntu 14.04 lts," Dell Inc. Dell Inc, may 2014. [Online]. Available: http://en.community.dell.com/techcenter/extras/m/white_papers/20439303

Fig. 10. Deploy bundle from Juju Charm Store [7].

```
juju deploy cs:bundle/wiki-simple-0
```