

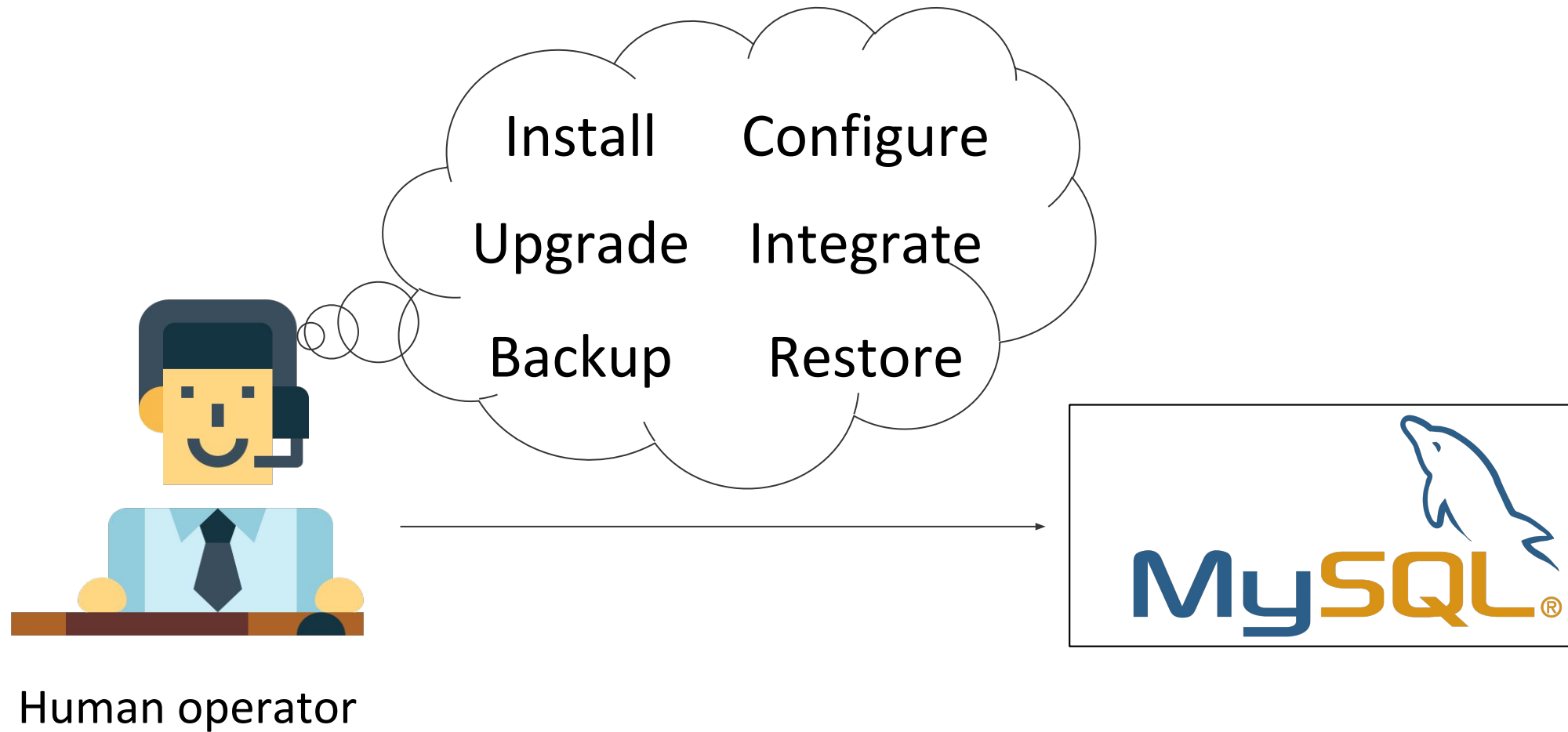
Open Source
MANO

Introduction to OSM Primitives

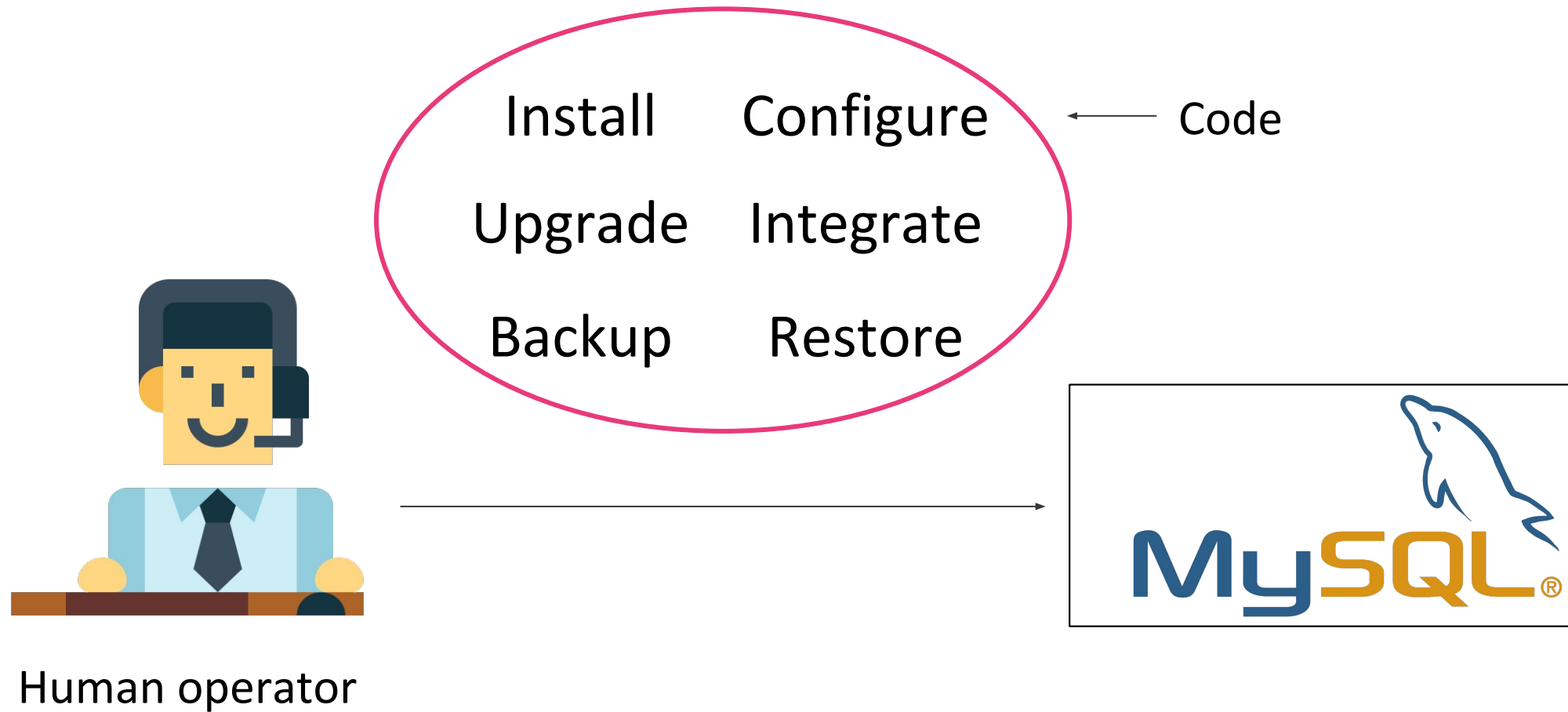
David Garcia (Canonical)

Primitives are actions exposed by the operator

Operator Pattern



Operator Pattern



Operator Pattern

Charm

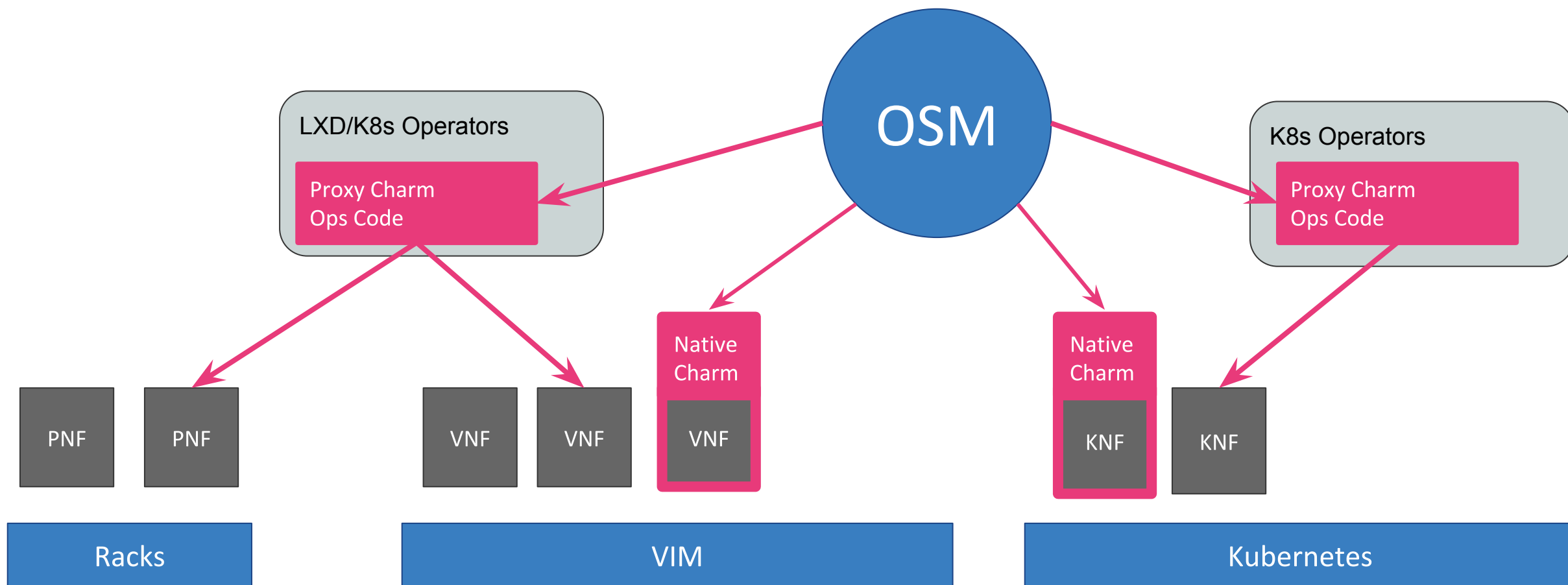


Install Configure
Upgrade Integrate
Backup Restore

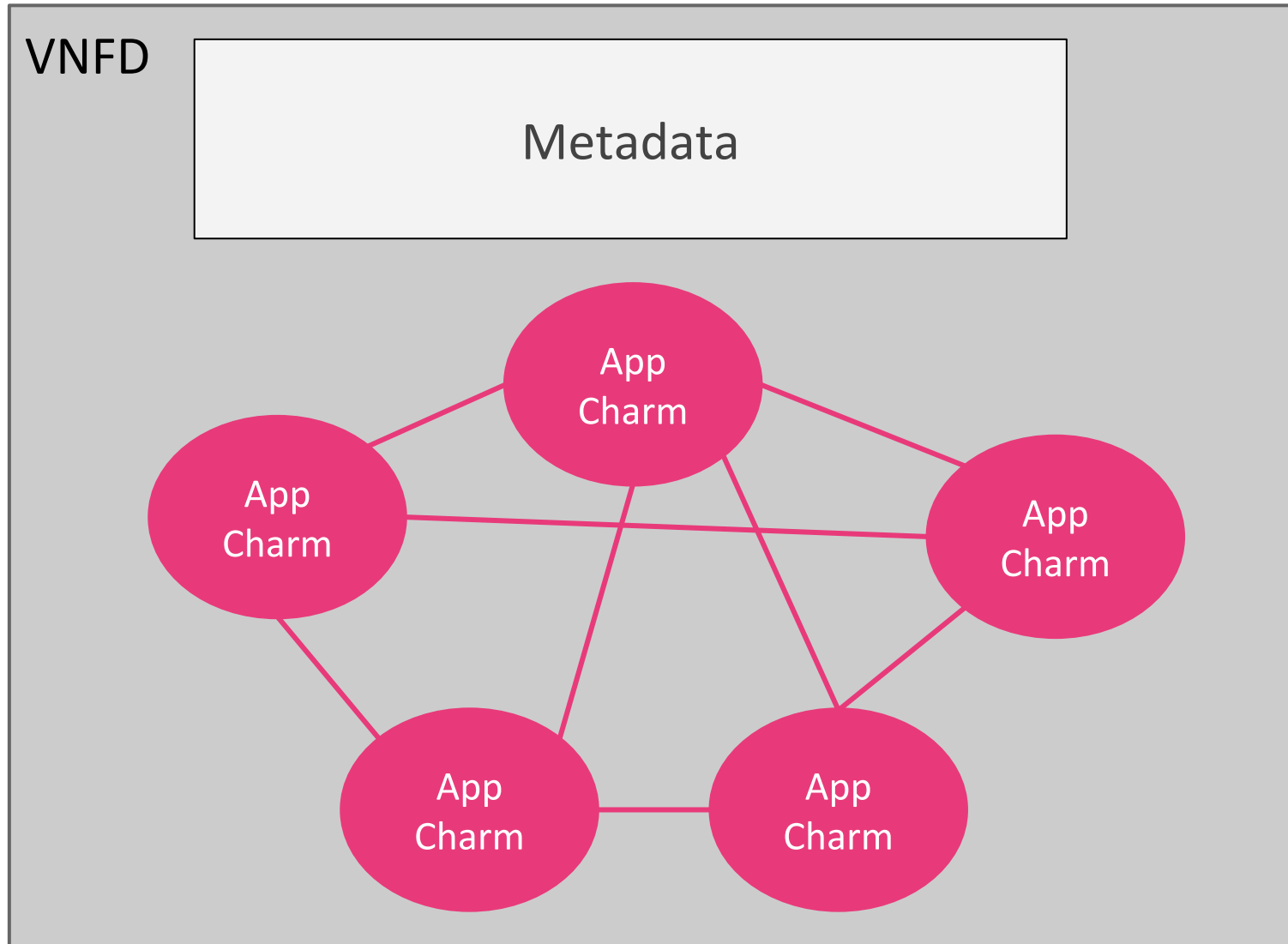


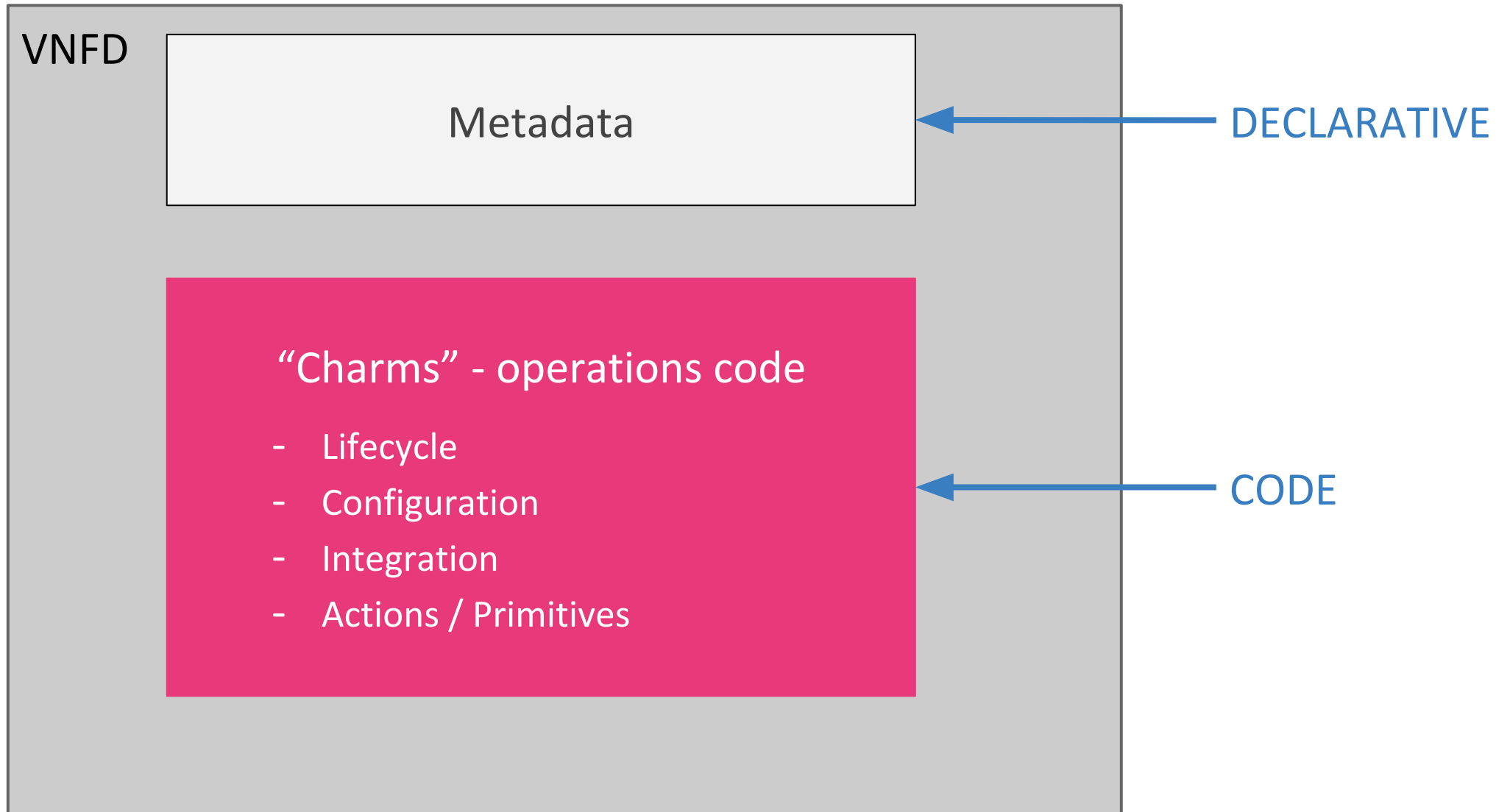
Charms are universal operators

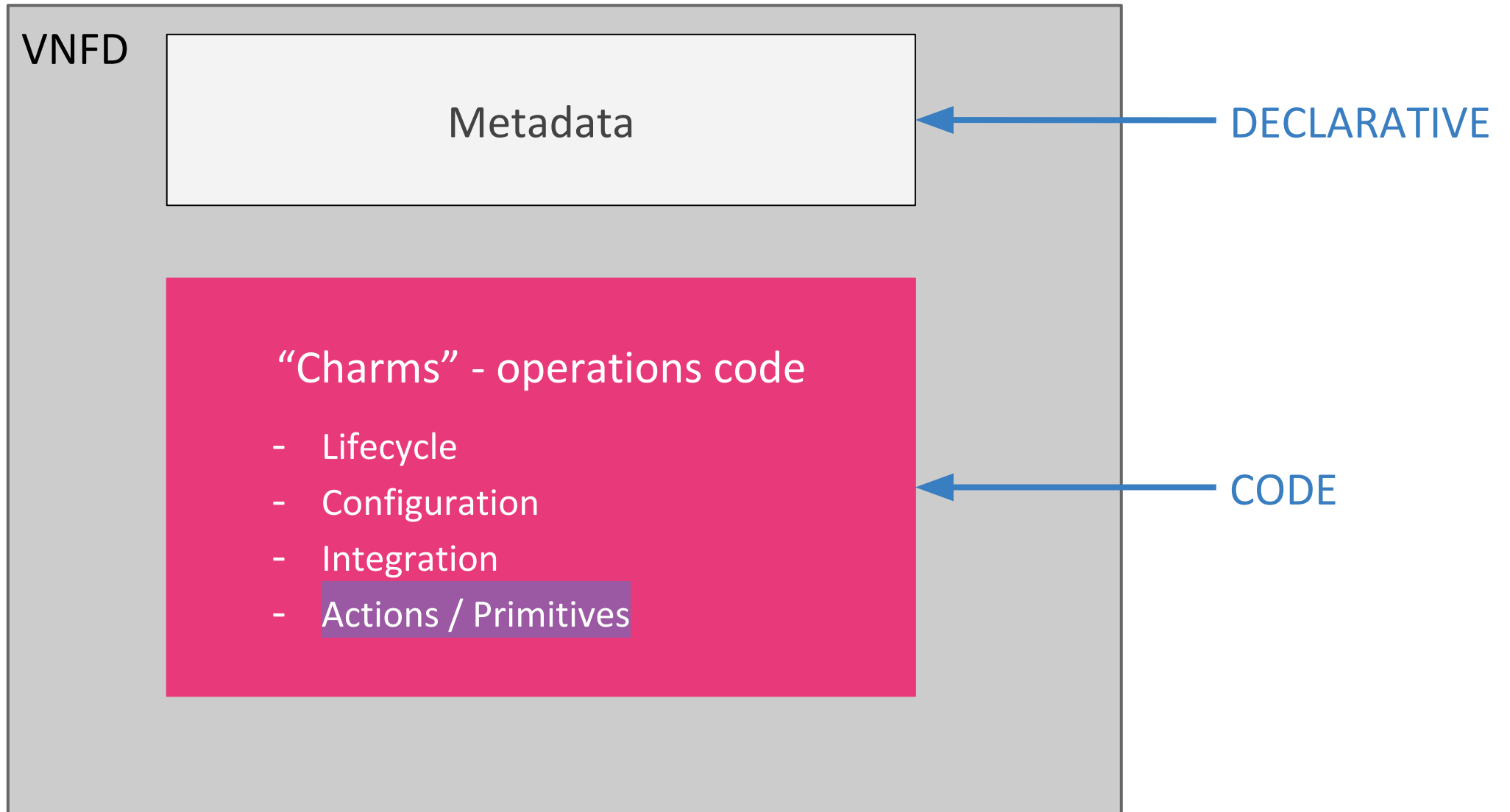
Reality is messy and mixed



One VNF is many apps and integration



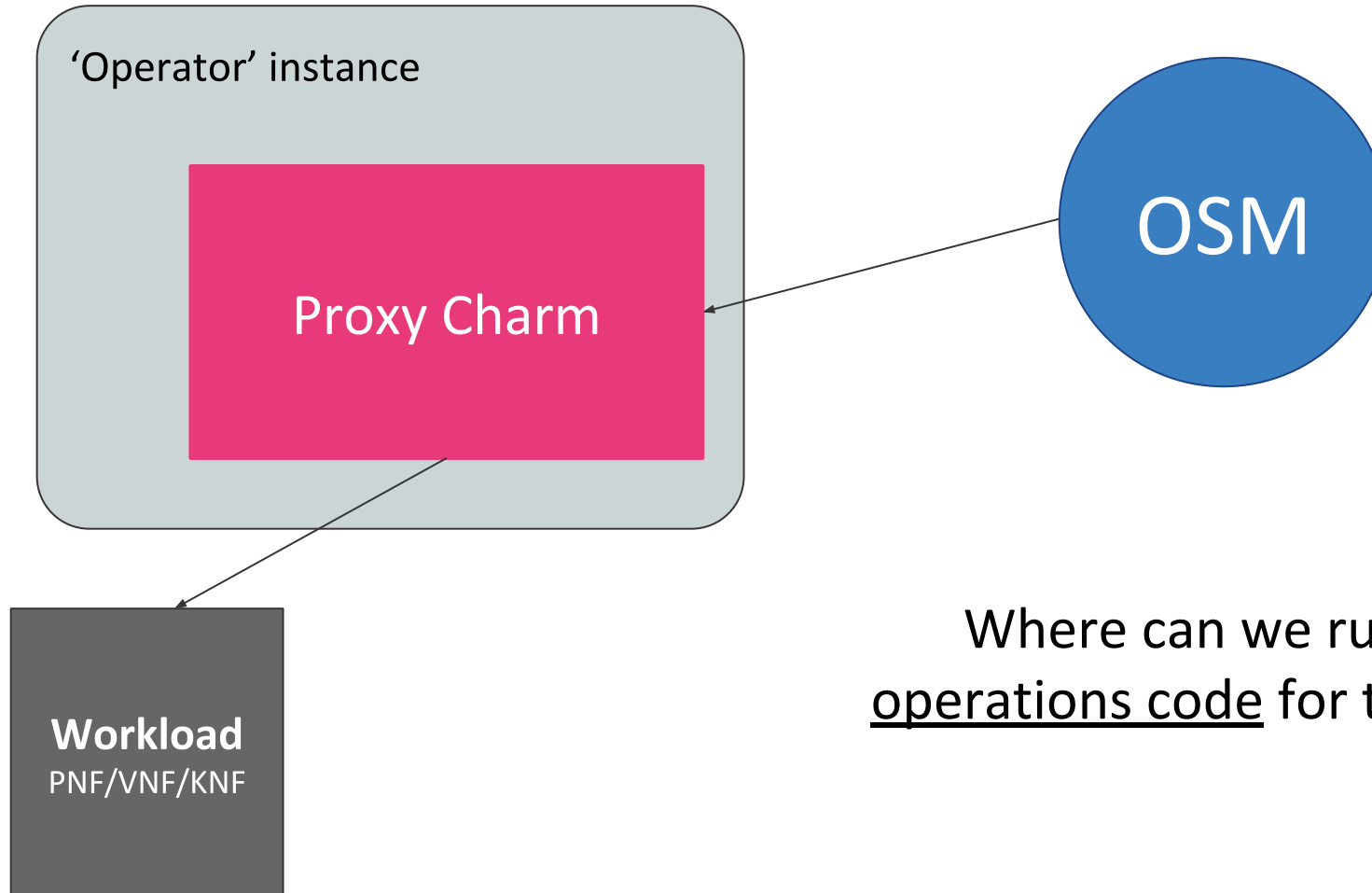




Primitives are actions exposed by the operator

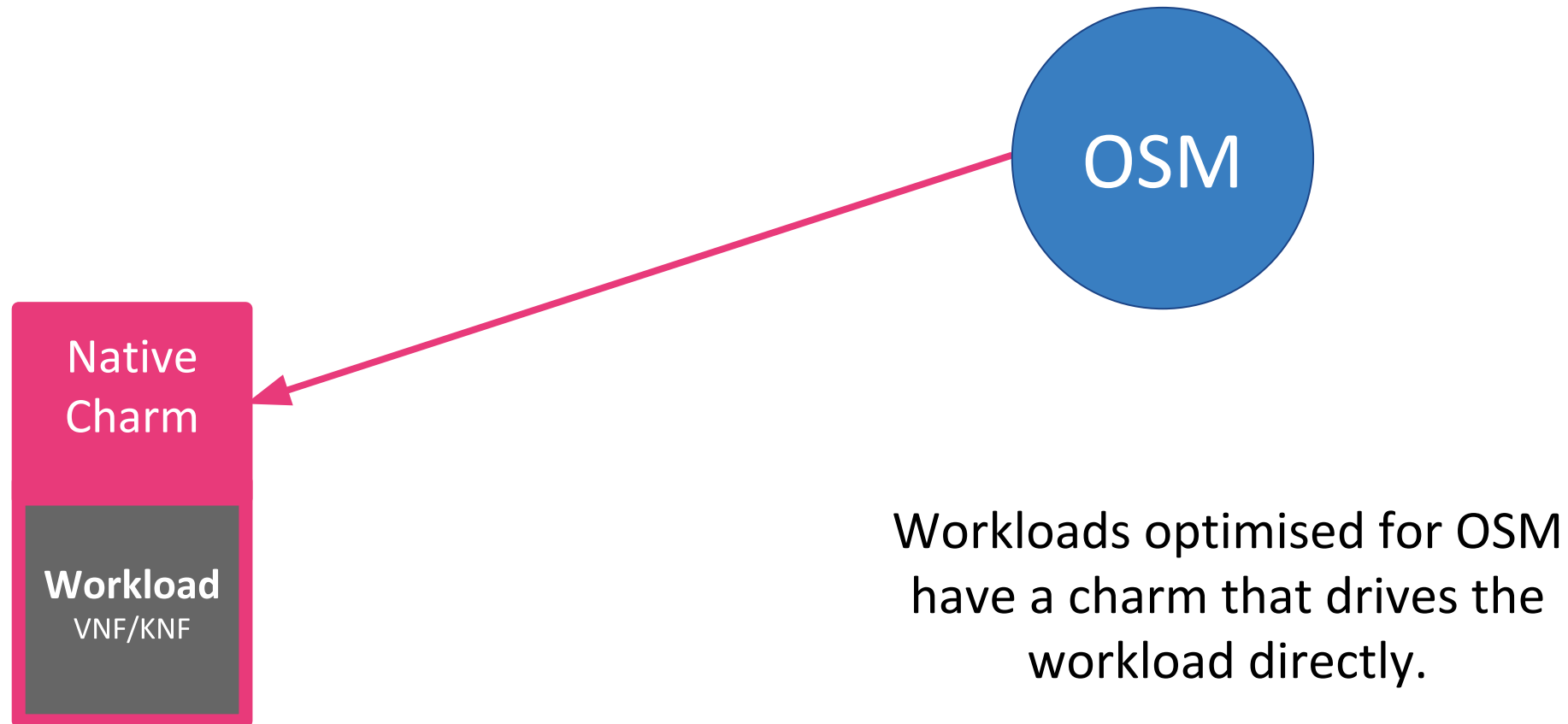
OSM Primitives are actions exposed by the Charm

Operating “proxy” workloads



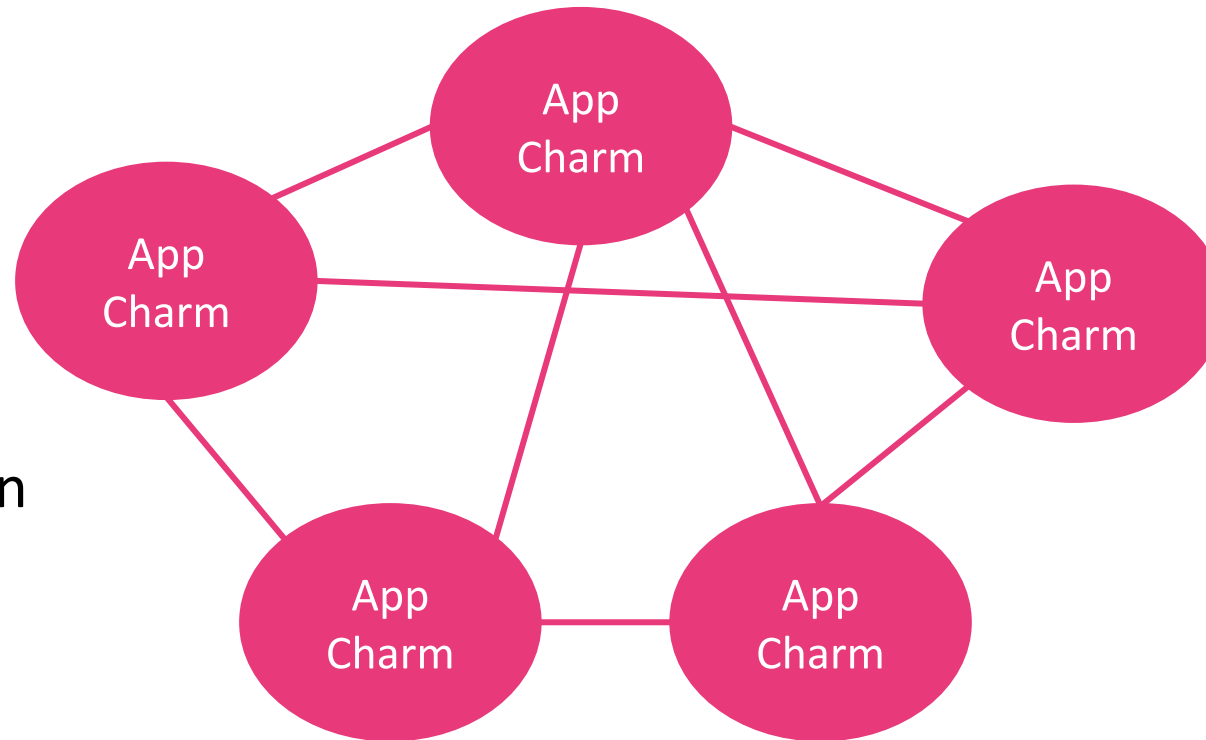
Where can we run our own
operations code for this workload?

Operating “native” workloads



Integration is first-class in the VNFd

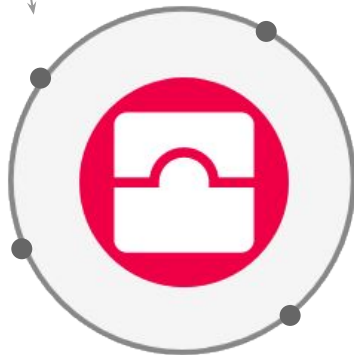
Lines of integration
in the VNFd



Charms declare **typed integration points**

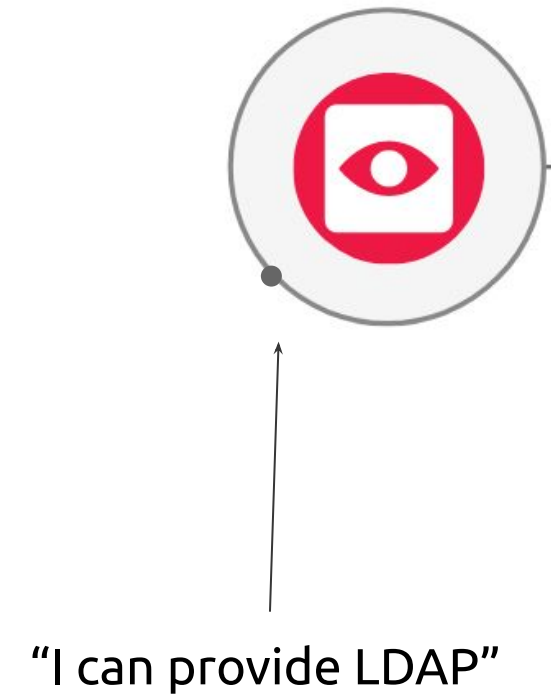
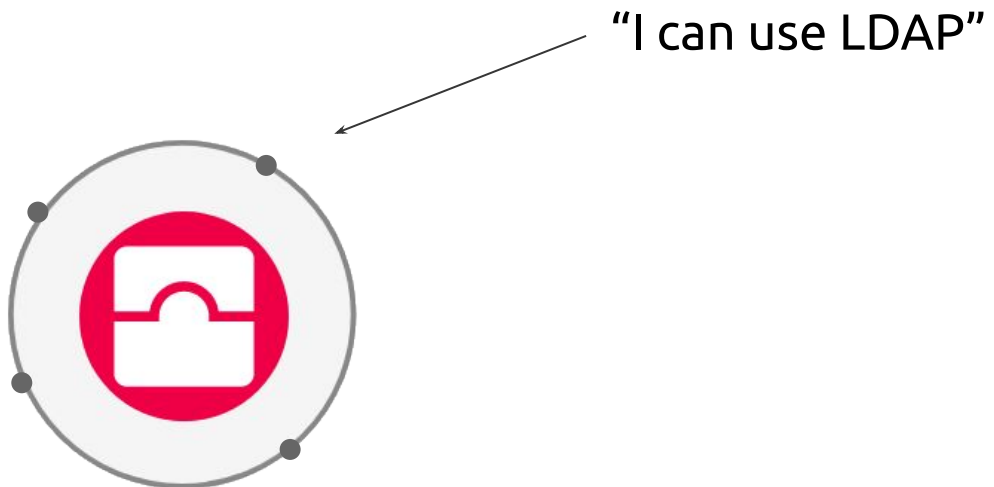
"I can use a MySQL database"

"I can use LDAP"

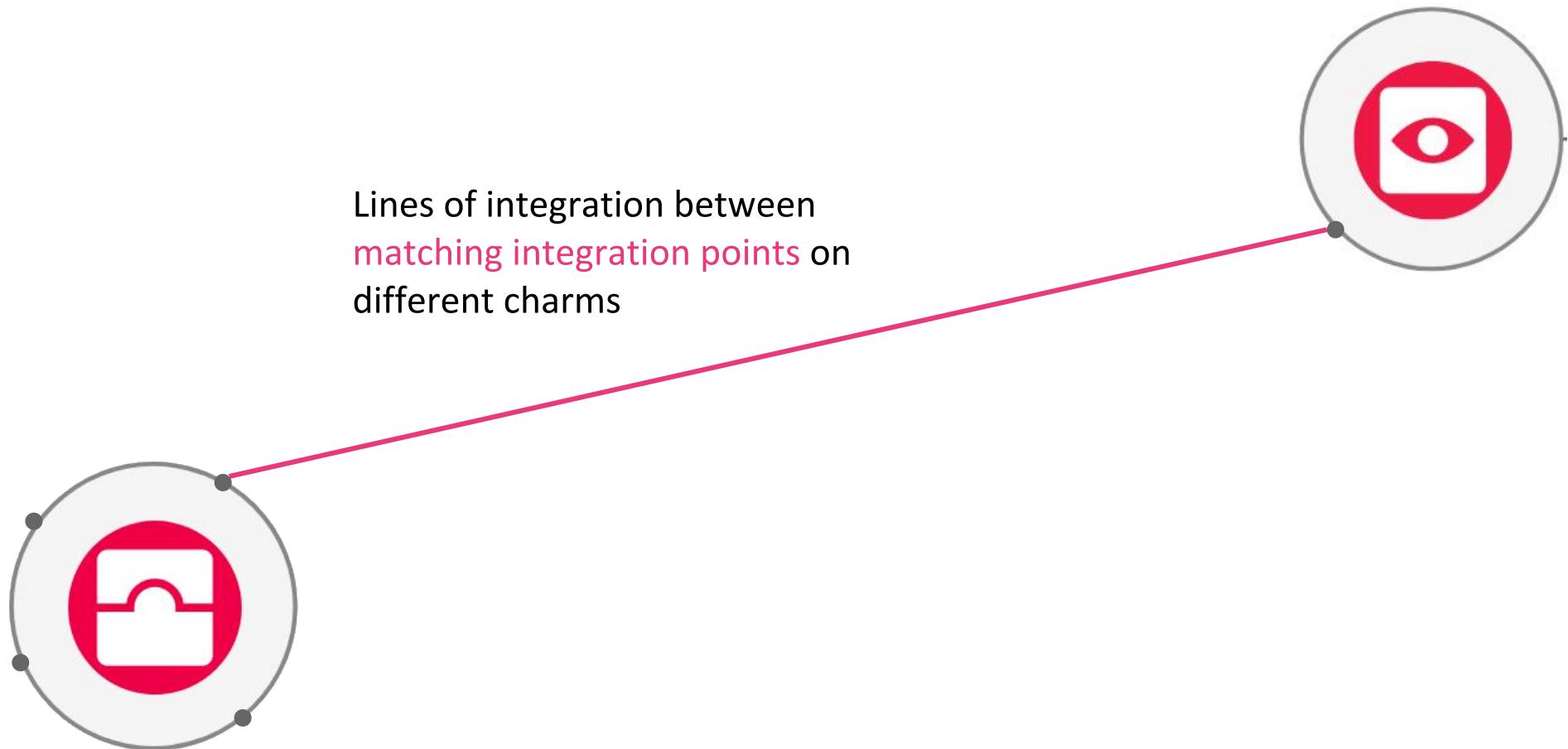


"I can send my logs to a syslog"

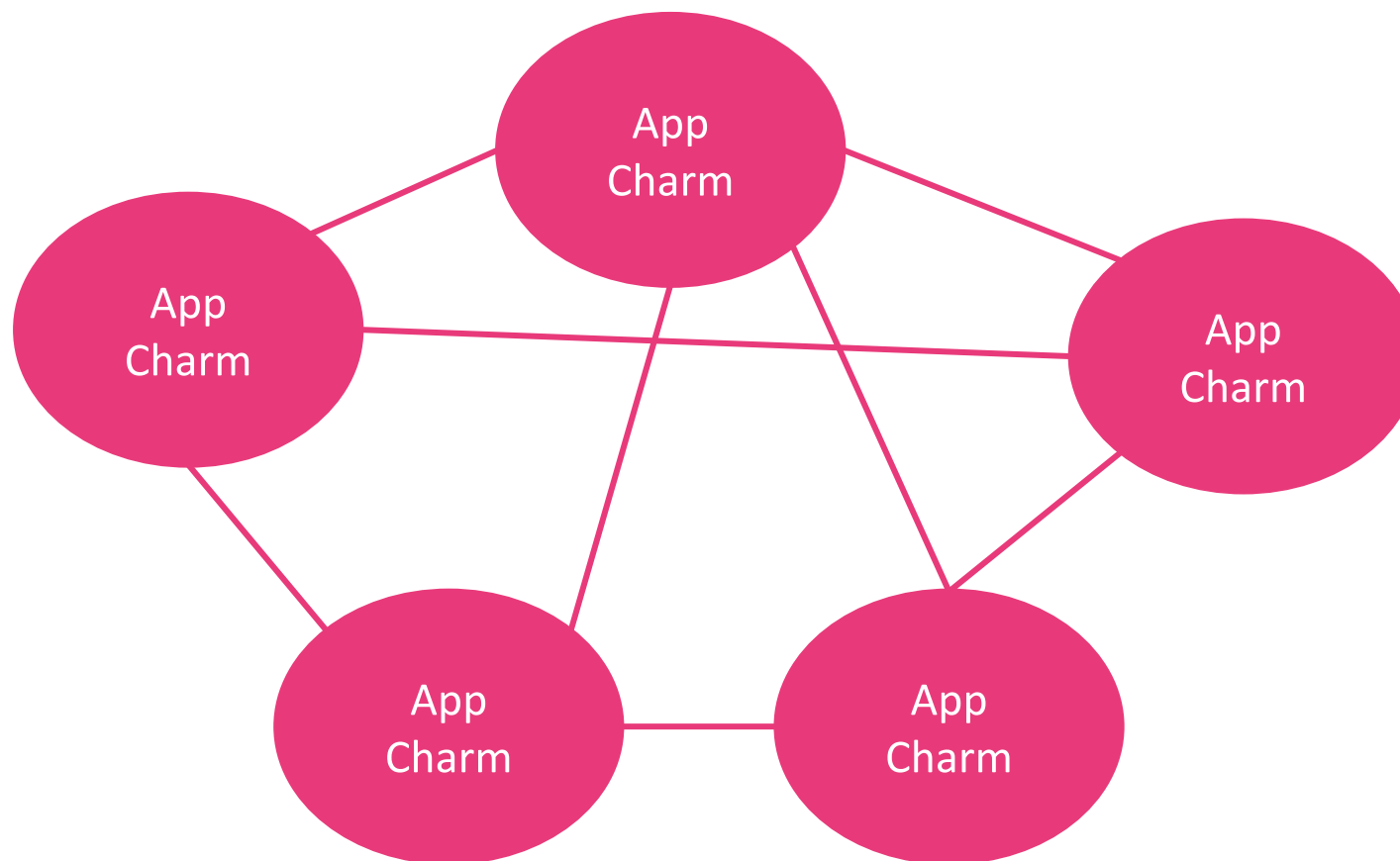
Charms declare **typed** integration points



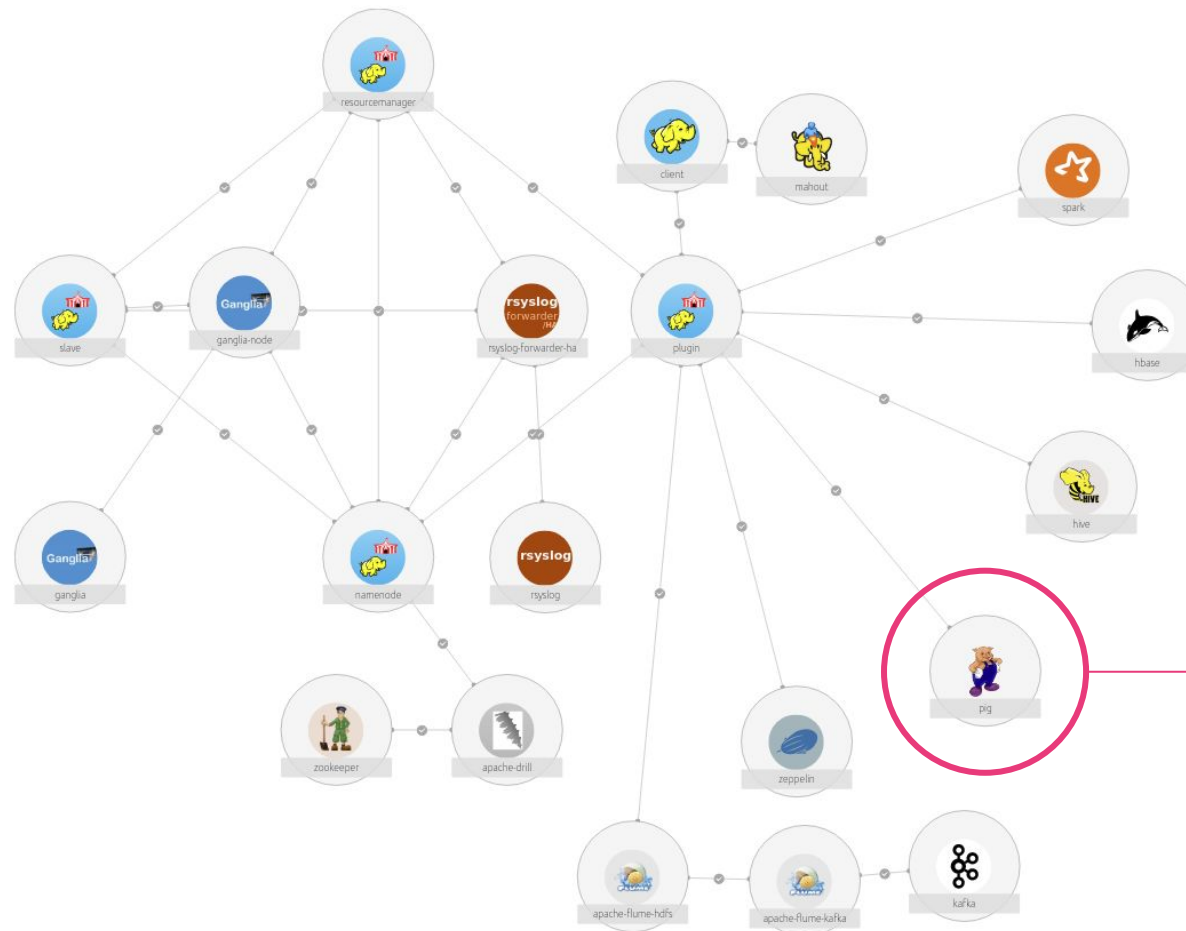
Matching integration points can be related



Composition gives complex integrations



VNFs can describe complex integrations



Lifecycle scripts
Config scripts
Integration scripts
Action scripts

OSM primitives are Charm Action scripts

- Backup
- Monitor
- Debug
- Add users, policies, rules, etc.
- Manage certificates, keys, etc.
- Rotate logs

Each 'primitive' is a **charm action** script that takes parameters and produces output.

Charms are packages of scripts to drive apps

Lifecycle scripts

- install
- config
- update
- remove
- scale

“Action” scripts are OSM Primitives

“action: backup”
“action: restore”
“action: scan-viruses”
“action: health-check”
“action: add-repo”
“action: ...”
“action: ...”
“action: ...”

Integration scripts

- relate-mysql
- relate-ldap
- relate-proxy
- relate-...

Charm

These are your
operations
primitives.



Charm describes Action parameters

Charm metadata describes the action parameters.

Each Action is a script, usually in Python or Bash.

```
addurl:
  description: "Add squid config"
  params:
    url:
      description: "URL that will be allowed"
      type: string
      default: ""
deleteurl:
  description: "Delete allowed URL squid config"
  params:
    url:
      description: "URL that will stop to be allowed"
      type: string
      default: ""
```

Charm Action script in bash

Actions can be written in bash for very simple cases.

```
#!/bin/bash

URL=`action-get url`

if ! grep -Fxq "http_access allow allowedurls"
/etc/squid/squid.conf
then
    sed -i '/^# And finally deny all .*/i http_access allow
allowedurls\n' /etc/squid/squid.conf
fi

sed -i "/^http_access allow allowedurls.*/i acl allowedurls
dstdomain \.${URL}" /etc/squid/squid.conf

kill -HUP `cat /var/run/squid.pid`
```

Charm Action script in python

It is common to write actions in Python using the standard Operator Framework.

```
def on_deleteurl_action(self, event):
    """Handle the deleteurl action."""
    url = event.params["url"]

    line_to_delete = "acl allowedurls dstdomain .{}".format(url)
    line_deleted = False

    with open("/etc/squid/squid.conf", "r") as f:
        lines = f.readlines()
    with open("/etc/squid/squid.conf", "w") as f:
        for line in lines:
            if line_to_delete not in line:
                f.write(line)
            else:
                line_deleted = True

    if line_deleted:
        event.set_results({"output": "URL deleted successfully"})
        subprocess.check_output(
            "kill -HUP `cat /var/run/squid.pid`", shell=True)
    else:
        event.fail("No URL was deleted")
```



Open Source
MANO

Juju Controller

Operator Lifecycle Manager

OSM Architecture

NBI
“Northbound Interface”

UI
“User Interface”

LCM
“Lifecycle Manager”

RO
“Resource Orchestrator”

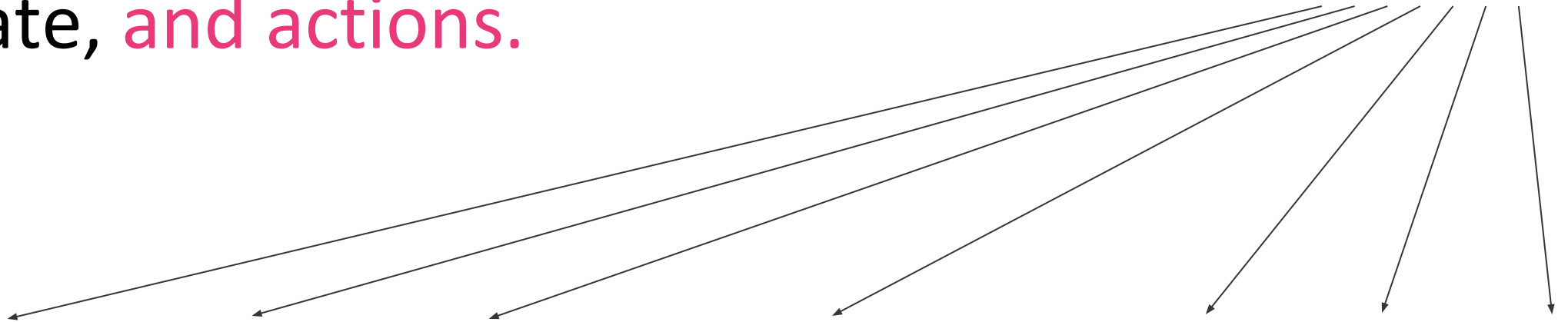
VCA
“VNF Configuration
Abstraction”



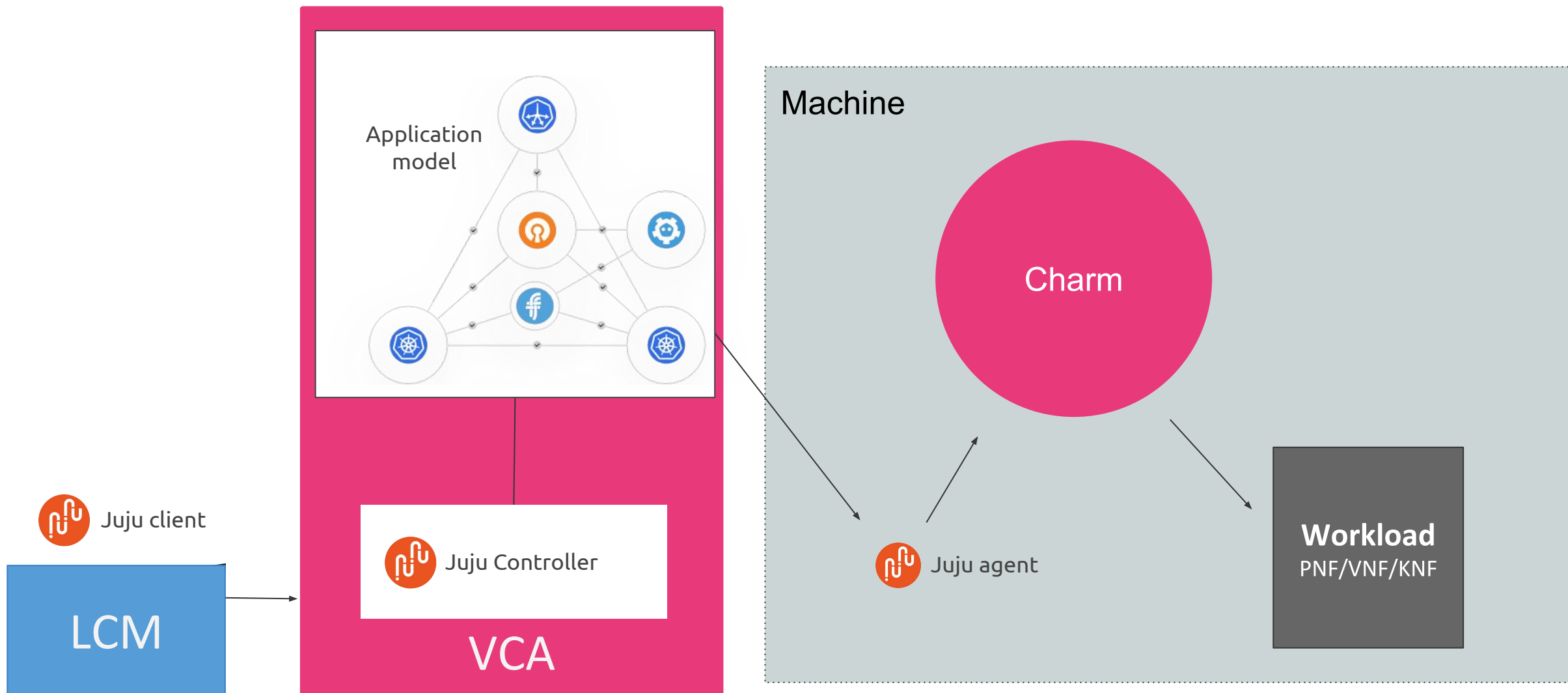
Juju drives application operations on machine and kubernetes substrates



Install, update, configure, scale,
integrate, and actions.



Juju architecture

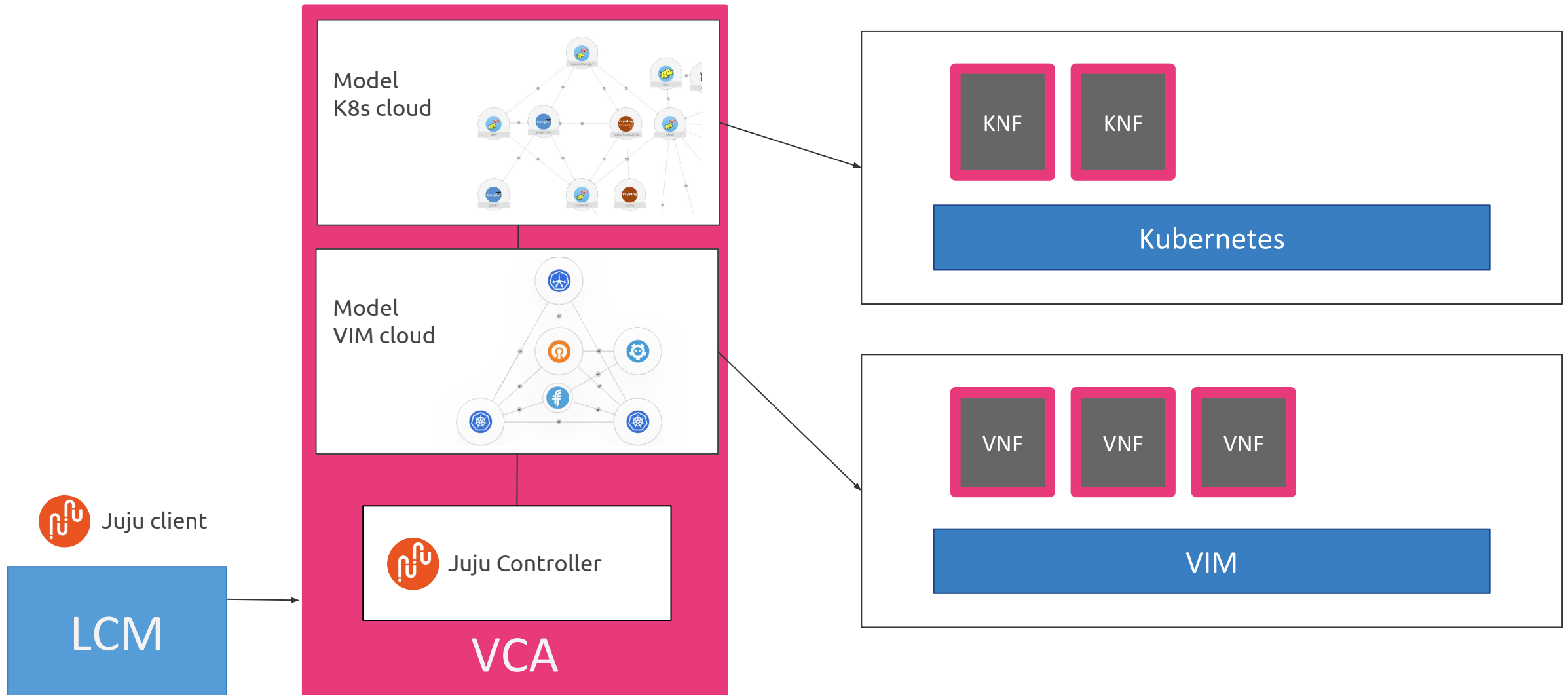




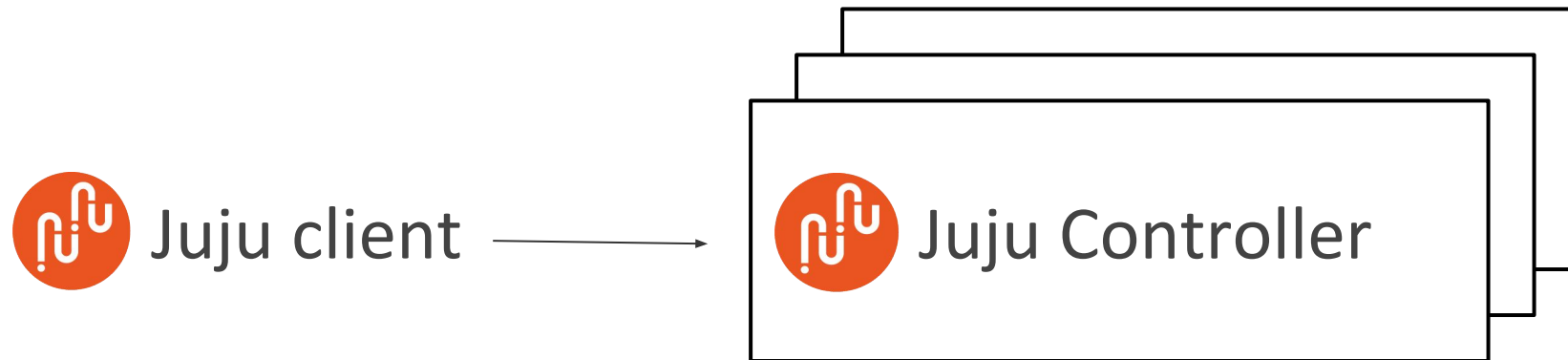
Open Source
MANO



VCA uses multiple models for scenario



VCA can be high availability



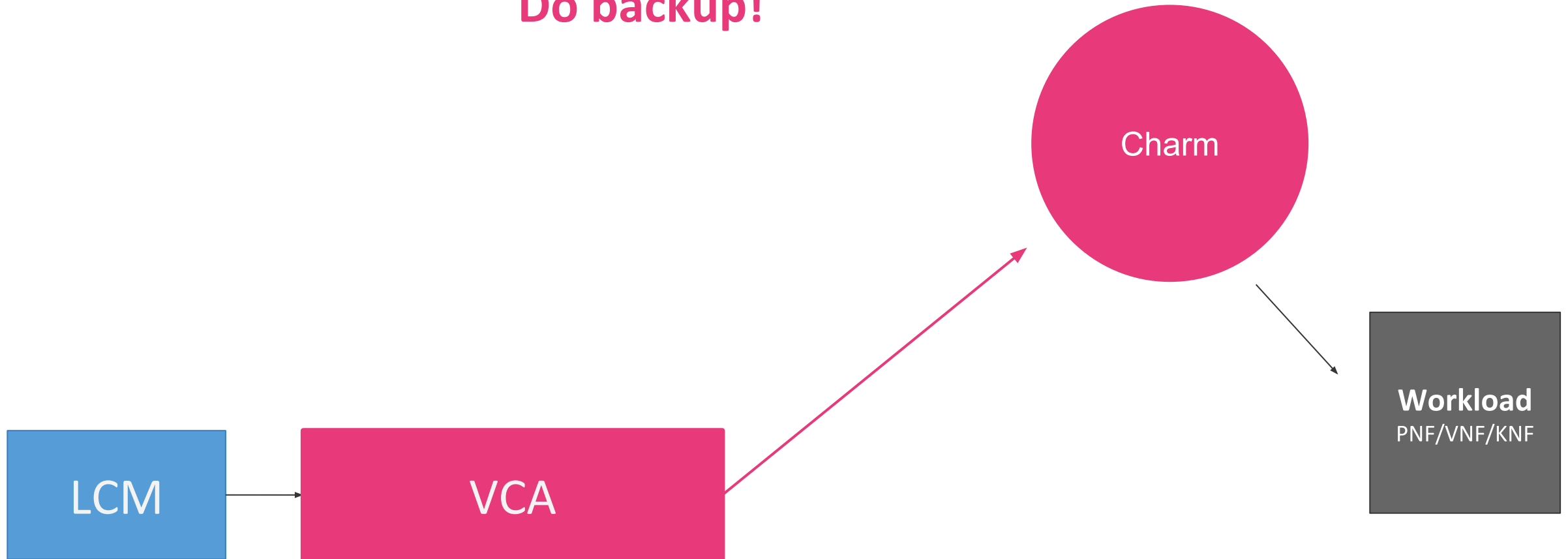
VCA coordinates all OSM Primitives

Configure

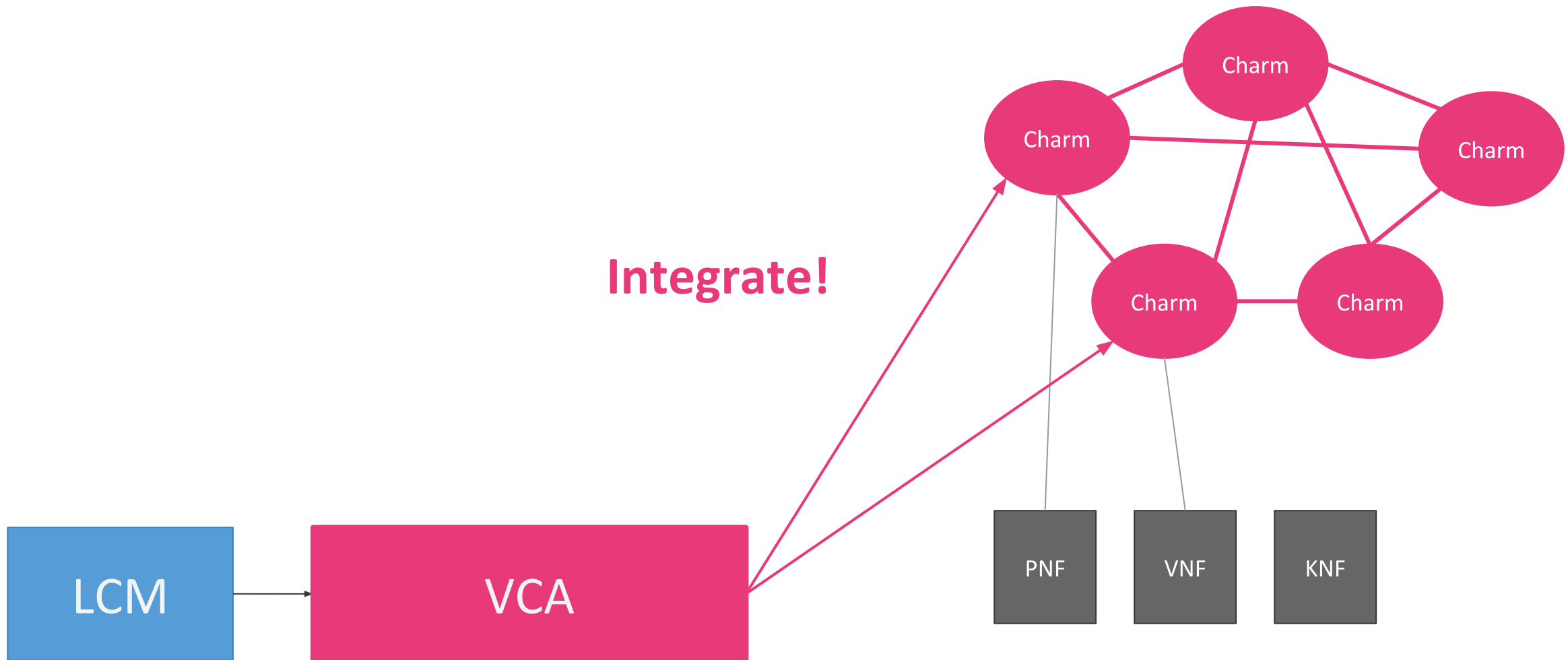
`https_proxy: xxx`
`ca_cert: yyyy`
`...`



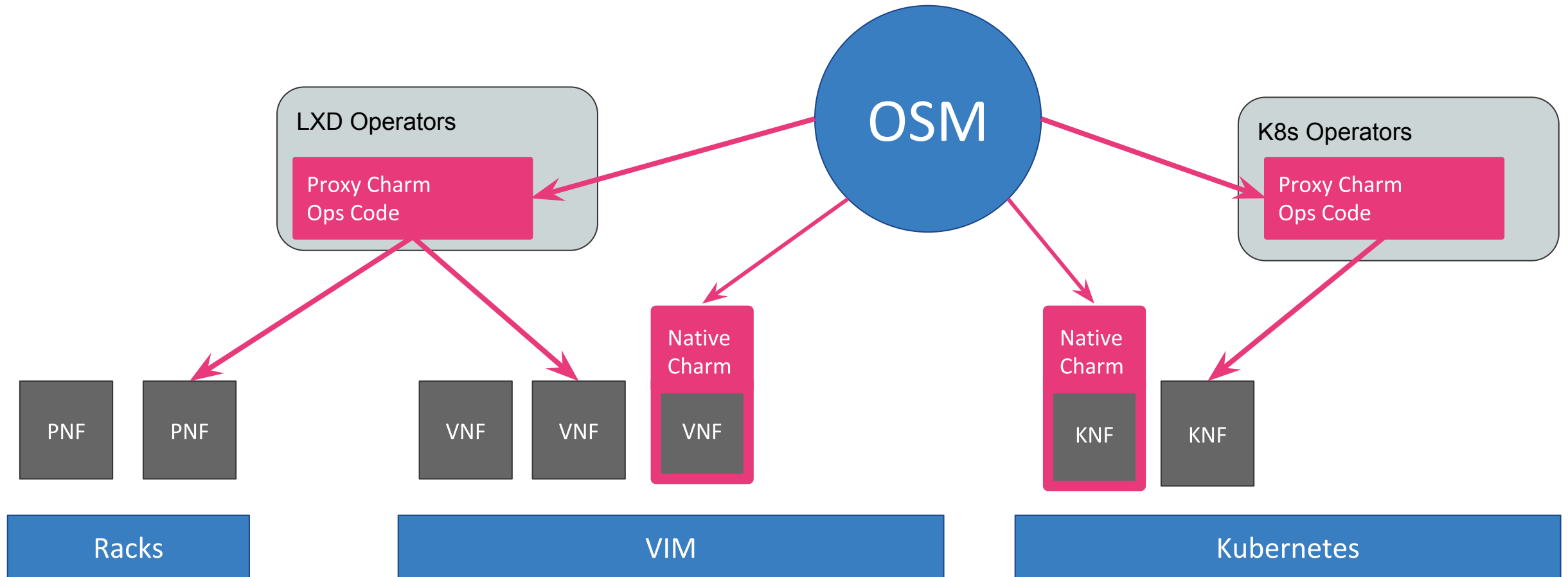
Do backup!

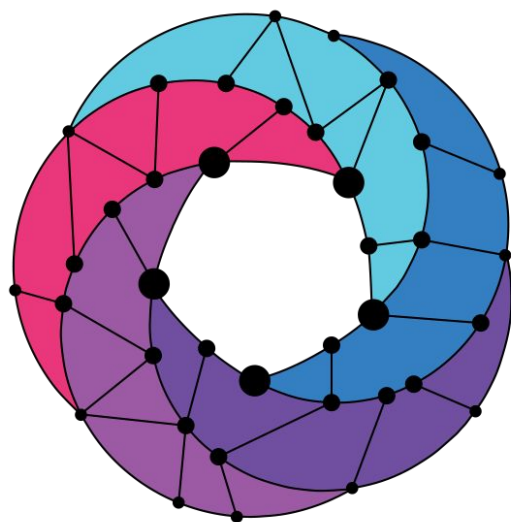


Integration



Reality is messy and mixed





Open Source MANO

Find us at:

osm.etsi.org
osm.etsi.org/wikipub