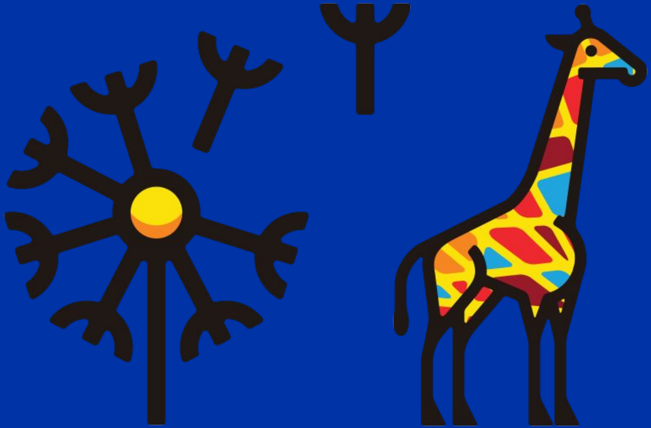
NOKIA

Advanced Fault Management with Vitrage and Mistral

Ifat Afek, Renat Akhmerov

November 2017



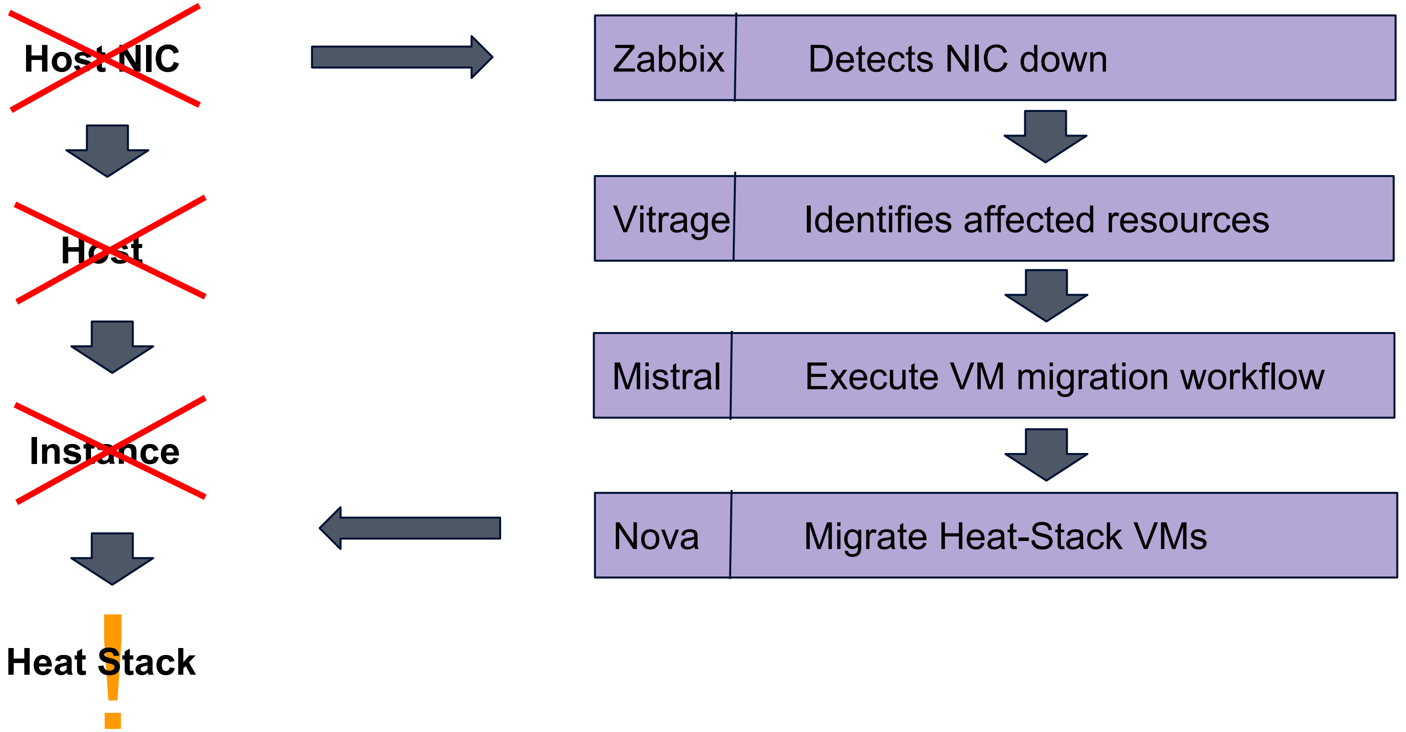
Fault Management in the Real World

Self-healing and fast recovery in real world cloud systems is challenging

- Failures happen in real distributed systems
- A single failure may affect many resources
- We can see symptoms but it’s hard to find the root cause
- Recovery might be complicated

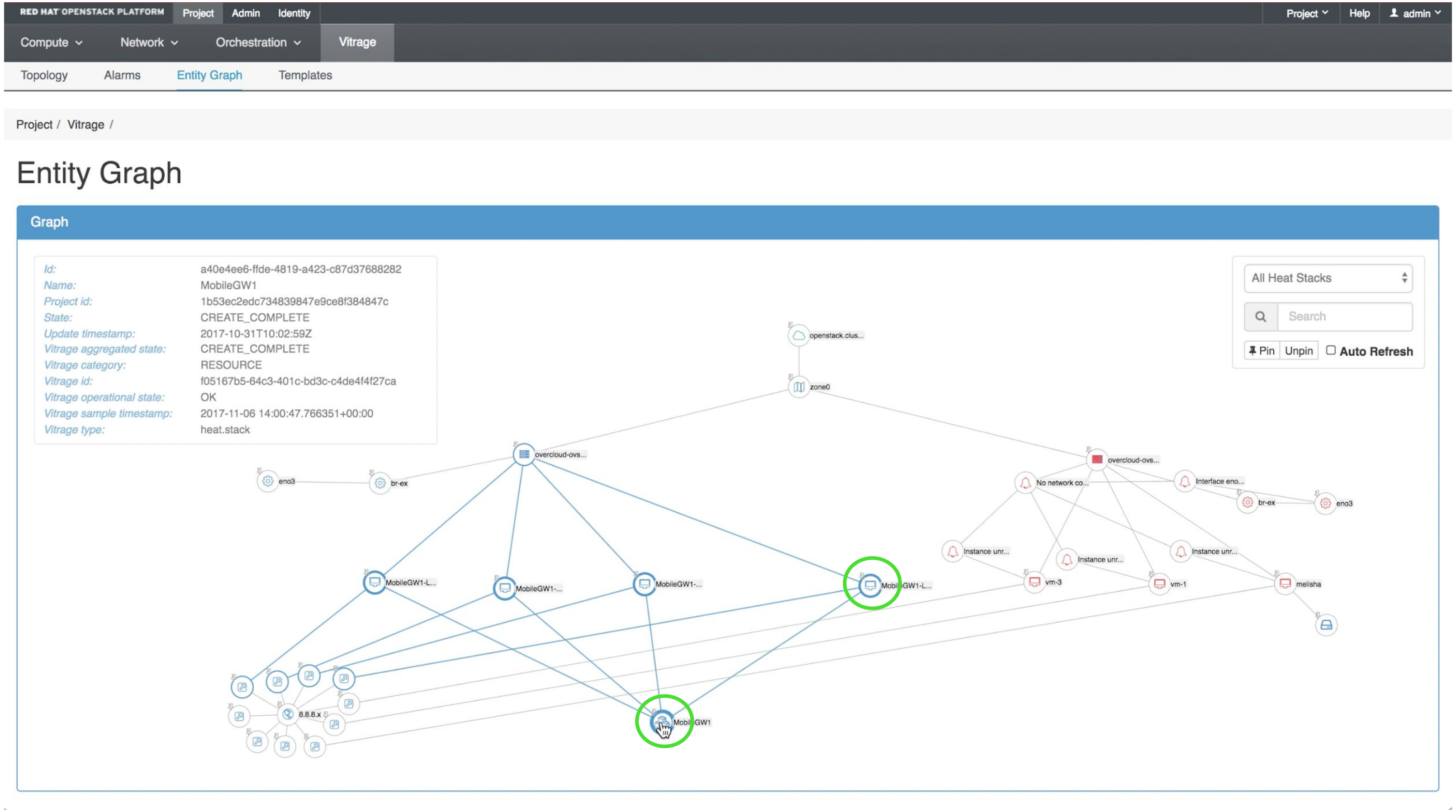
Vitrage and Mistral can solve this!

Demo


Host NIC
Host
Instance
Heat Stack
Zabbix
Detects NIC down
Vitrage
Identifies affected resources
Mistral
Execute VM migration workflow
Nova
Migrate Heat-Stack VMs

Demo

After VM Migration



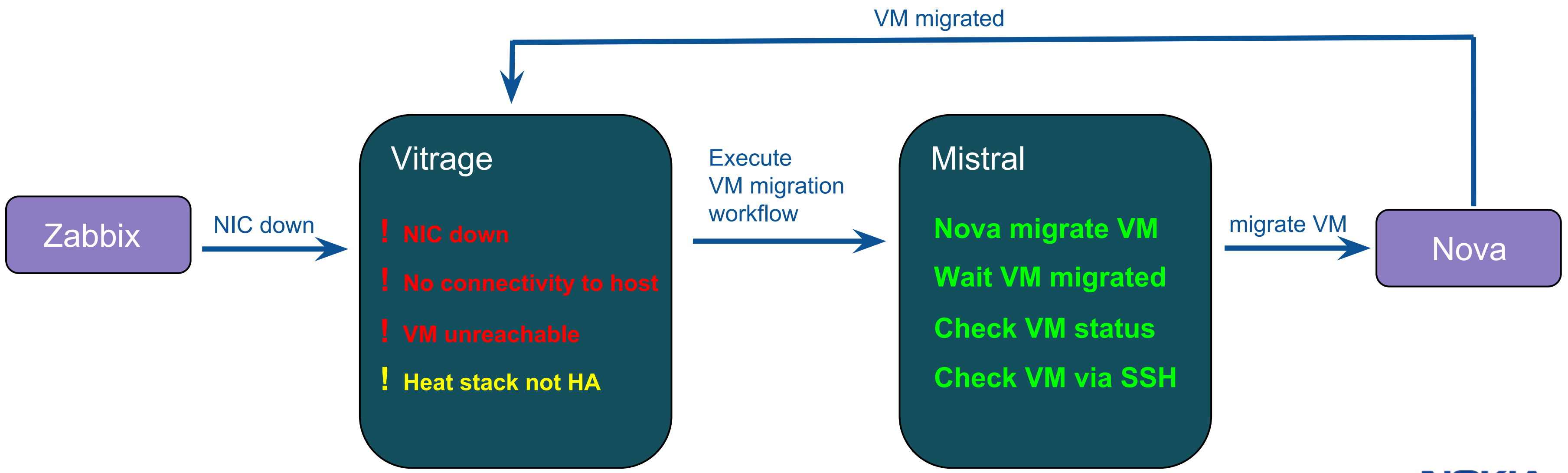
Entity Graph

Id:	a40e4ee6-ffde-4819-a423-c87d37688282
Name:	MobileGW1
Project id:	1b53ec2edc734839847e9ce8f384847c
State:	CREATE_COMPLETE
Update timestamp:	2017-10-31T10:02:59Z
Vitrage aggregated state:	CREATE_COMPLETE
Vitrage category:	RESOURCE
Vitrage id:	f05167b5-64c3-401c-bd3c-c4de4f4f27ca
Vitrage operational state:	OK
Vitrage sample timestamp:	2017-11-06 14:00:47.766351+00:00
Vitrage type:	heat.stack

How did the Demo work?

Vitrage - provides **insights** about the system
Mistral - a **workflow** service

Vitrage + Mistral => **pinpoint the problems and take corrective actions!**



Using Vitrage to Provide Insights

Vitrage - an official OpenStack project for Root Cause Analysis

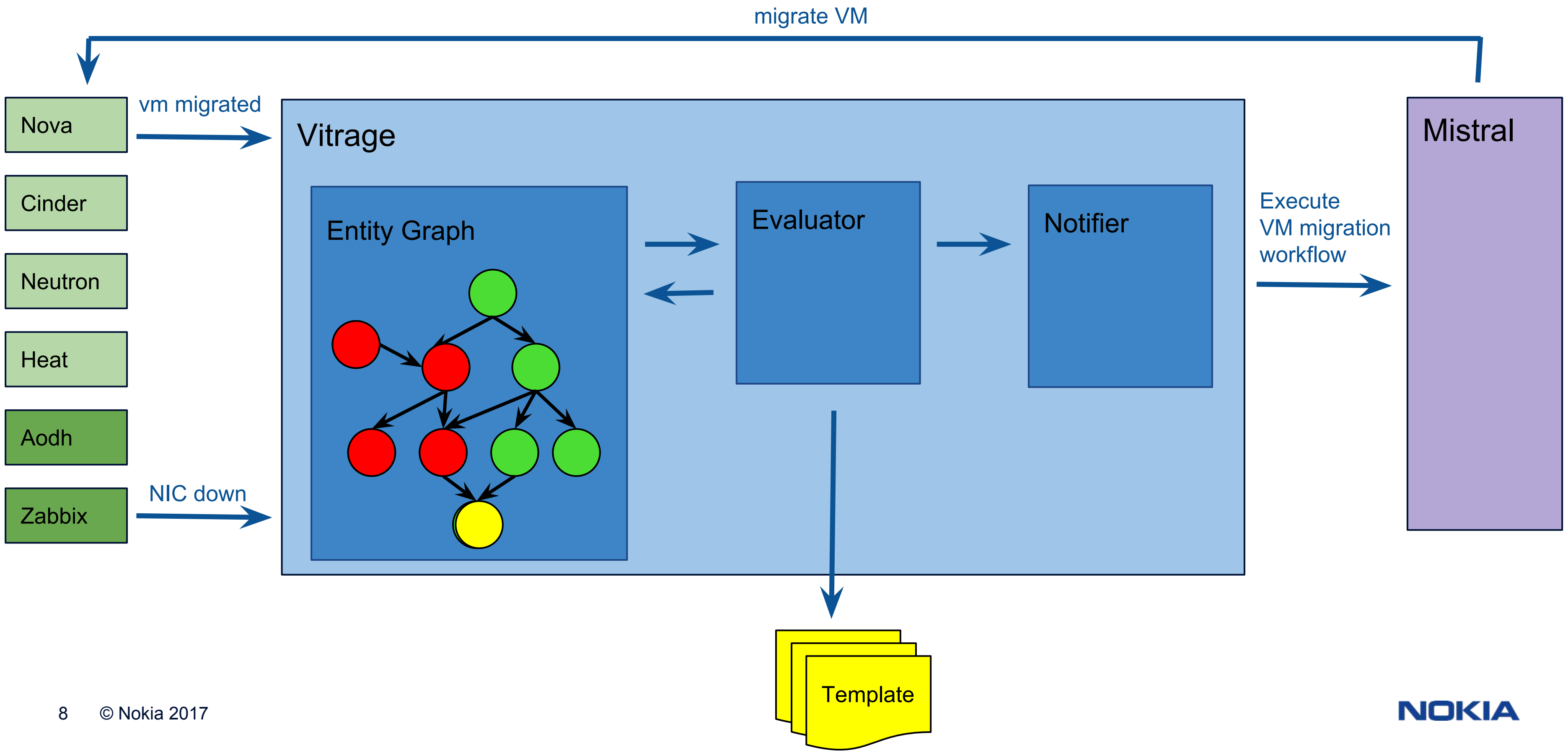
Vitrage Functions:

- Root Cause Analysis – understand what causes faults to occur
- Deduced alarms and states – raising alarms and modifying states based on system insights
- Holistic & complete view of the system

VITRAGE

an OpenStack Community Project

Using Vitrage to Provide Insights and Trigger Mistral Workflow



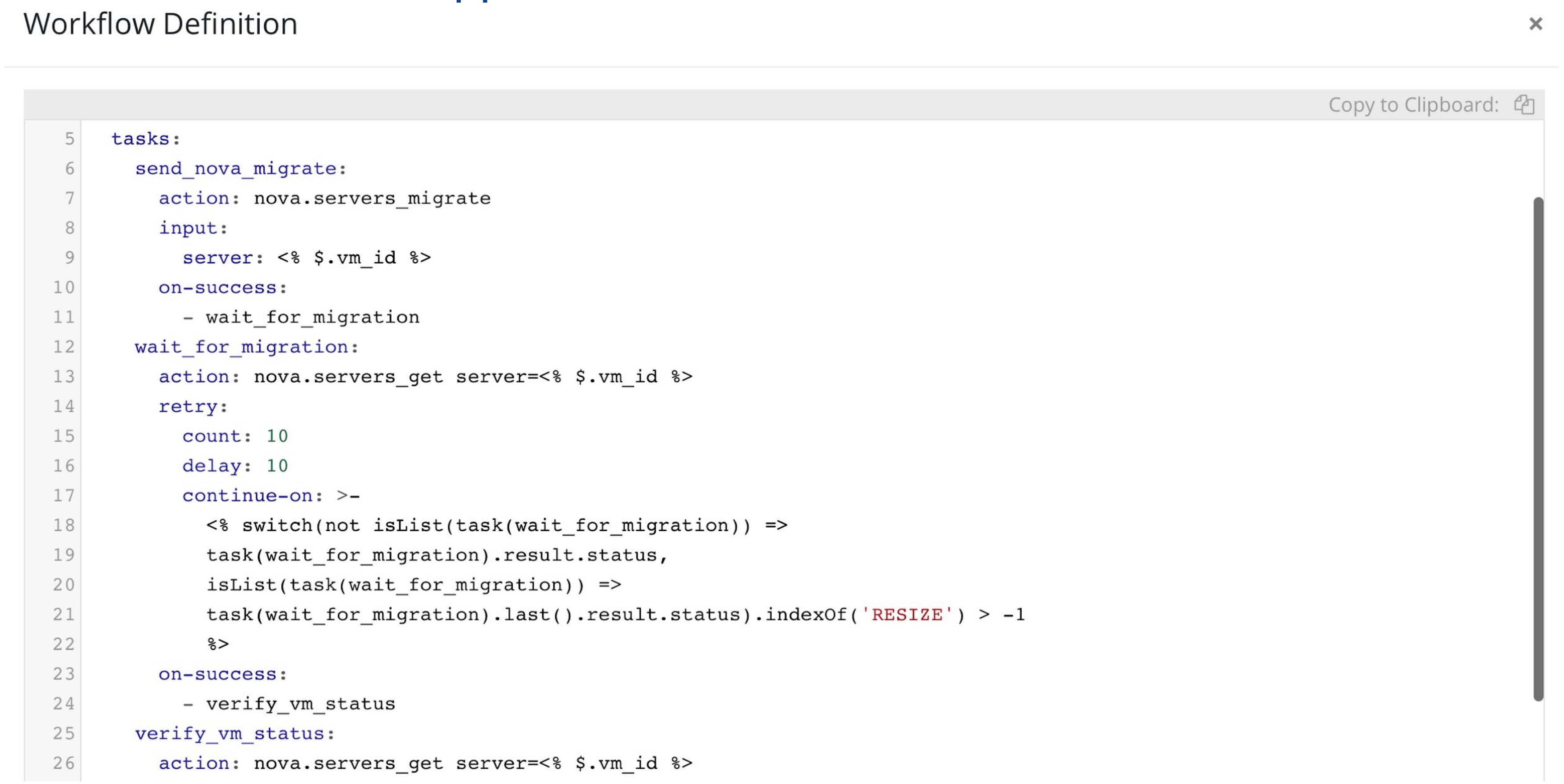
Mistral Workflow Service

- Highly mature OpenStack project
- Highly Available and Horizontally Scalable service to run and manage workflows
- YAML-based Workflow Language
- REST API
- Workflow execution history
- Flexible data manipulation with YAQL & Jinja
- Various task execution policies (retry, timeout etc.)
- “Glue” for OpenStack services that provides a single execution and auth context

MISTRAL

an OpenStack Community Project

Mistral Workflow Snippet



Workflow Definition

Copy to Clipboard:

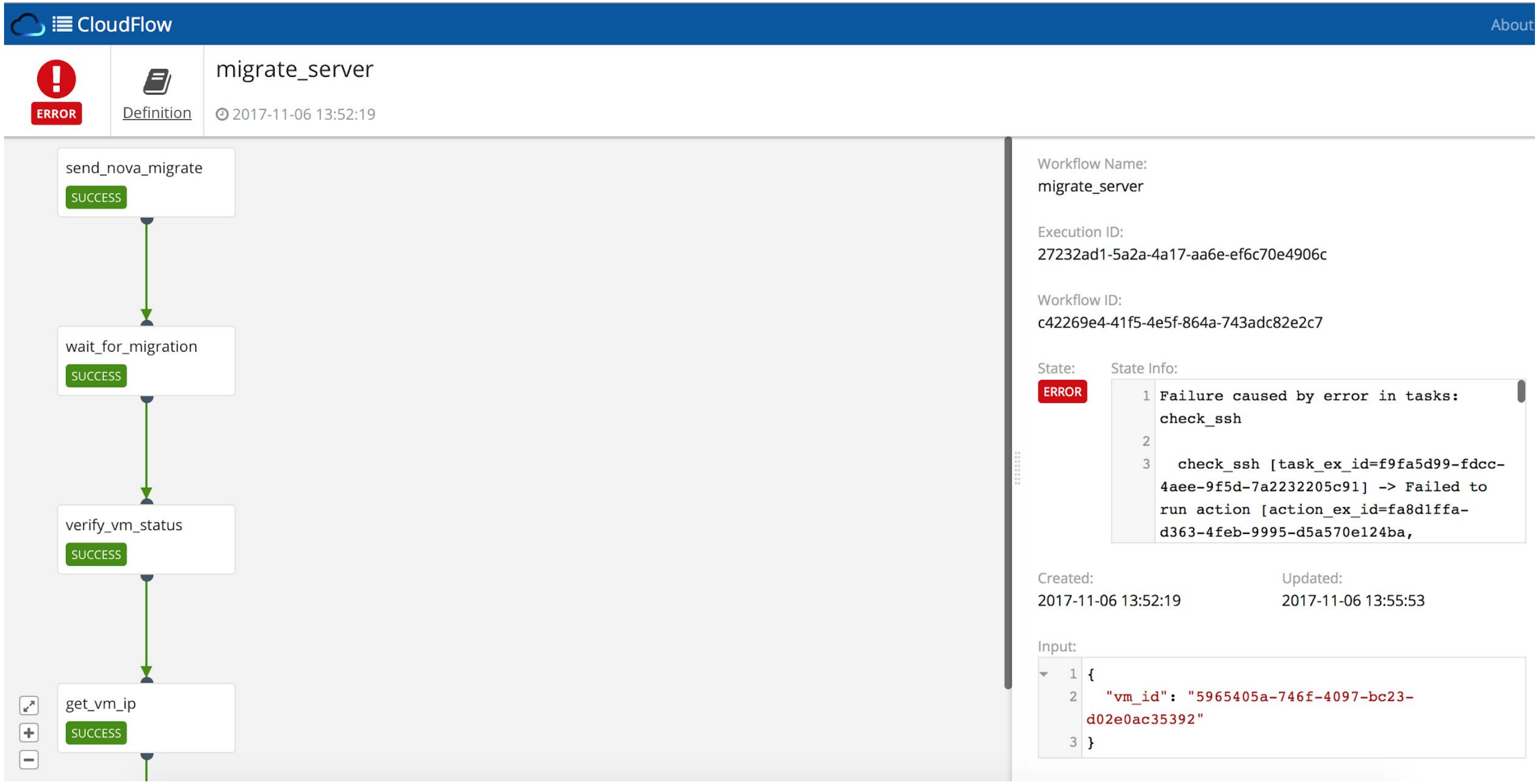
```
tasks:
  send_nova_migrate:
    action: nova.servers_migrate
    input:
      server: <% $.vm_id %>
    on-success:
      - wait_for_migration
  wait_for_migration:
    action: nova.servers_get server=<% $.vm_id %>
    retry:
      count: 10
      delay: 10
      continue-on: >-
        <% switch(not isList(task(wait_for_migration)) =>
        task(wait_for_migration).result.status,
        isList(task(wait_for_migration)) =>
        task(wait_for_migration).last().result.status).indexOf('RESIZE') > -1
        %>
    on-success:
      - verify_vm_status
  verify_vm_status:
    action: nova.servers_get server=<% $.vm_id %>
```

CloudFlow - standalone GUI for Mistral

What is CloudFlow?

- Open Source project - https://github.com/nokia/CloudFlow
- Visualization of Mistral workflow executions
- Fully reflects Mistral API Data Model
  - “join” operations
  - Data collections
  - Sub-workflows
  - Data flow
- Draws Mistral workflow executions of any complexity

The Demo workflow in CloudFlow



migrate_server

2017-11-06 13:52:19

Workflow Name:
migrate_server

Execution ID:
27232ad1-5a2a-4a17-aa6e-ef6c70e4906c

Workflow ID:
c42269e4-41f5-4e5f-864a-743adc82e2c7

State: ERROR

State Info:

```
Failure caused by error in tasks: check_ssh

  check_ssh [task_ex_id=f9fa5d99-fdcc-4aee-9f5d-7a2232205c91] -> Failed to run action [action_ex_id=fa8d1ffa-d363-4feb-9995-d5a570e124ba,
```

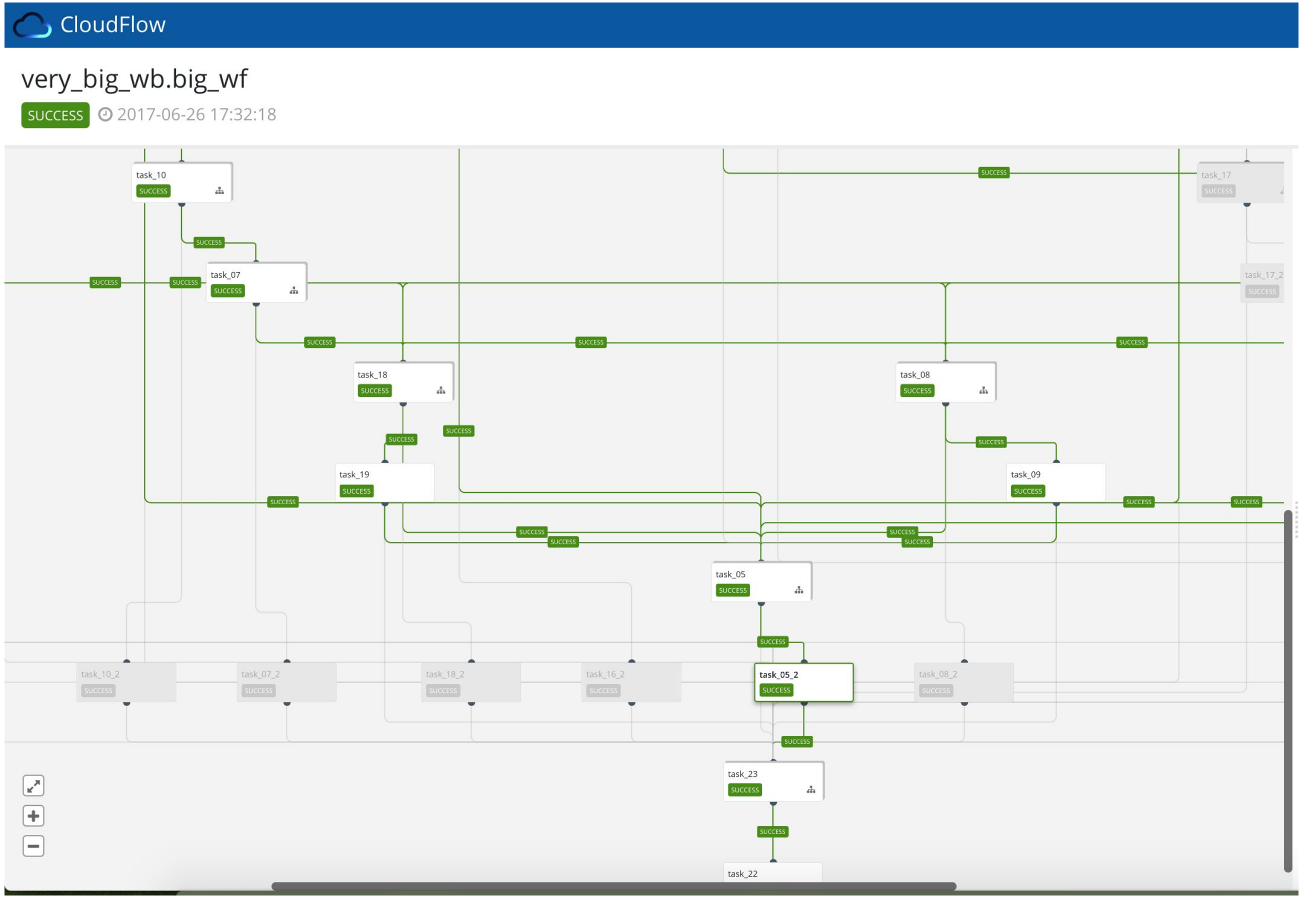
Created: 2017-11-06 13:52:19

Updated: 2017-11-06 13:55:53

Input:

```
{
  "vm_id": "5965405a-746f-4097-bc23-d02e0ac35392"
}
```

CloudFlow can also do this!



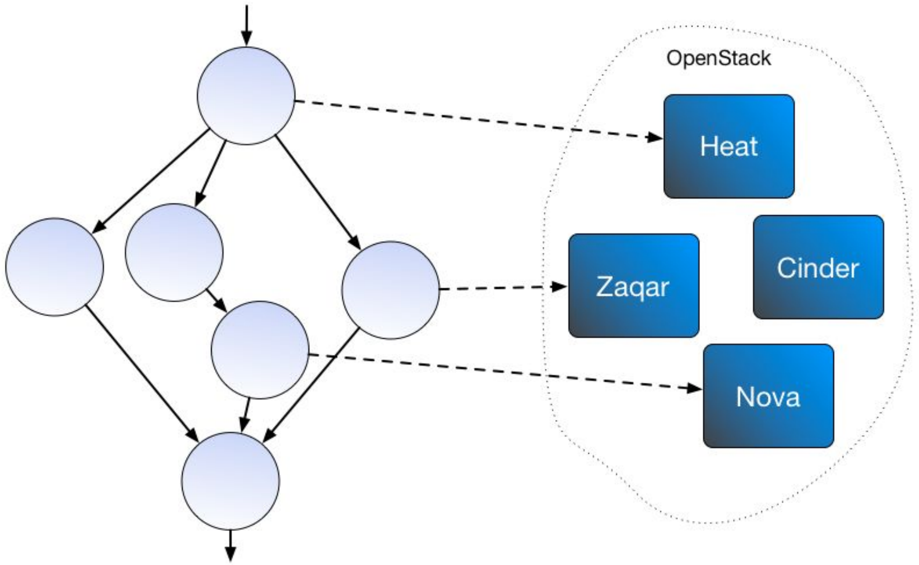
Thanks to the CloudFlow author :)

Guy Shaanan

Full Stack Web Developer
at Nokia

Why Mistral?

- Workflows can be updated dynamically
    - No need to reconfigure or redeploy the environment
- We can associate any custom logic with any event
    - Mitigate a failure
    - Notify an operator
    - Provision resources
    - Scale up / Scale down
- Using workflows is safe
- Workflows are stateful



Other reasons to use Mistral

- Workflows are scalable
- Workflows can be visualized
- High Availability
- Parallelism and asynchrony
- Reliable data transfer in a distributed environment
- Human intervention into a running process
- Recovery from errors
- Easy to combine different languages

NOKIA

Contributors are Welcome!

Vitrage

- wiki page: https://wiki.openstack.org/wiki/Vitrage
- IRC Channel: #openstack-vitrage
- OpenStack Dev mailing list – use [vitrage] tag

Mistral

- Documentation: https://docs.openstack.org/mistral
- IRC Channel: #openstack-mistral
- OpenStack Dev mailing list – use [mistral] tag

NOKIA