NTT Communications
Transform. Transcend.

OpenStack Operation Under a Multi-tenant and Multi-customer Public Cloud Environment

2016年7月7日
NTT Communications　堀田孝司

Transform your business, transcend expectations with our technologically advanced solutions.

Outline

Introduction

NTT Communications

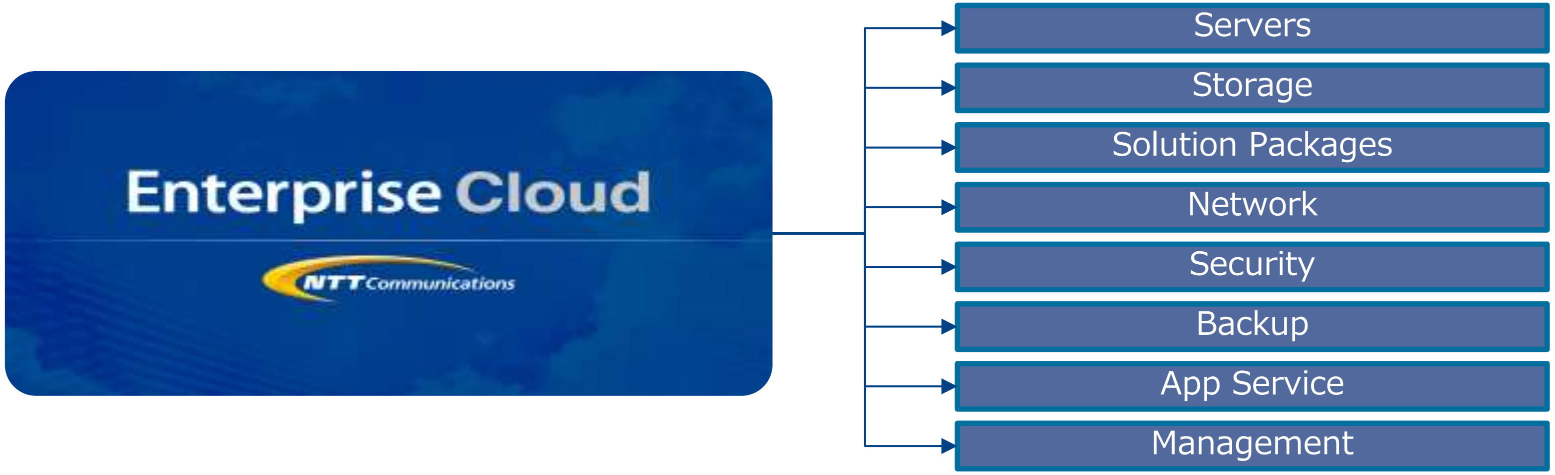
- Headquarters in Tokyo, Japan

- NTT Communications is one of the leading cloud providers in Japan
- One of the biggest datacenter operators in the world

- Services
  - Datacenter (140+ countries/regions)
  - VPN (196 countries/regions)
  - Global Tier1 Internet Backbone (Top 3 worldwide)
  - Worldwide Marine Cable (Top 10 worldwide)
  - IaaS/PaaS services worldwide , etc.

In what service do we use OpenStack?

- NTT Communications Enterprise Cloud
    - IaaS/PaaS/Managed Cloud

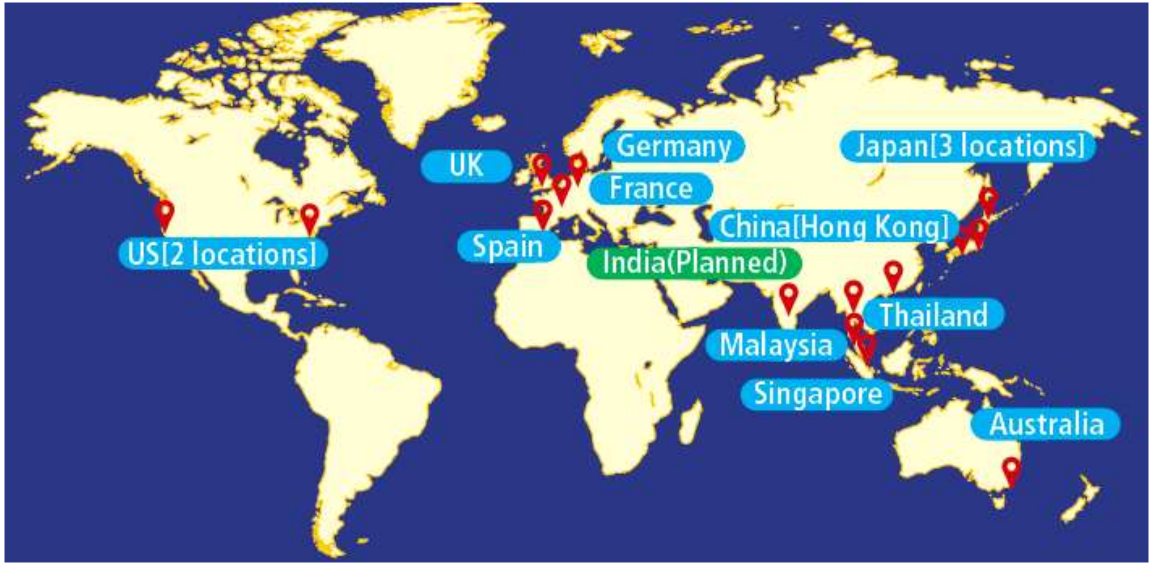


NTT Communications’ Enterprise Cloud in the Global Market

- Available in 14 different regions (+1 planned)

- Global Affiliates
  - NTT America
  - NTT Europe
  - NTT Singapore
  - NTT Com Asia
  - NTT Com ICT, etc.

- Multiple support teams
- Multiple languages



Why use OpenStack?

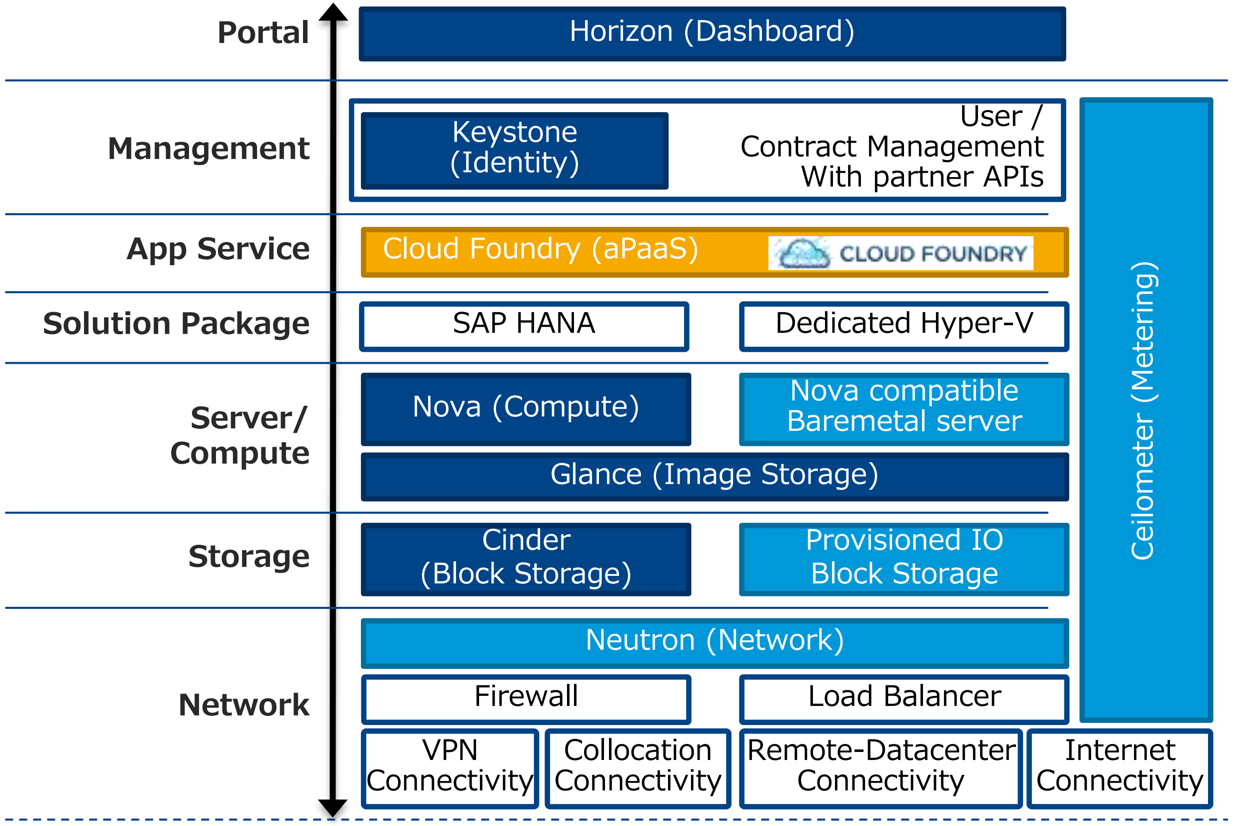
- NTT Communications Enterprise Cloud utilizes OpenStack because
  - Open-source
  - Expanding and active developer community

- Version used: JUNO

JUNO
THE TENTH OPENSTACK RELEASE

http://www.openstack.org/software/juno

OpenStack Components / Others in Enterprise Cloud



■ : OpenStack Component

■ : Original Component with Compatible/Partially compatible API

□ : Original Component

■ : Other OSS

Business Background for Our Service

- Main target users: Enterprise users
- There are gaps between the OpenStack community version and what the user wants as an IaaS
  - High Availability (HA) function for Virtual Machines
  - Multi-customer / Multi-tenant Environment

Requirements for Our Service

Requirements for Our Service

1. To support both traditional IT and cloud-native IT
2. Multi-customer/multi-tenant environment

Requirements for Our Service

1. **To support both traditional IT and cloud-native IT**
2. Multi-customer/multi-tenant environment

Requirement: To support both traditional IT and cloud-native IT

Pet Model vs Cattle Model

CC-BY-SA3.0 By Daniel Schwen
http://creativecommons.org/licenses/by-sa/3.0/

- Unique and given names
- Cared for
- Nursed back to health when sick

- Identical to one another / cannot tell apart
- Easily replaced

Requirement: To support both traditional IT and cloud-native IT

Pet Model

- Traditional IT
- Currently legacy apps cannot yet be easily replaced

- Case: If one VM goes down it will impact the end-user greatly

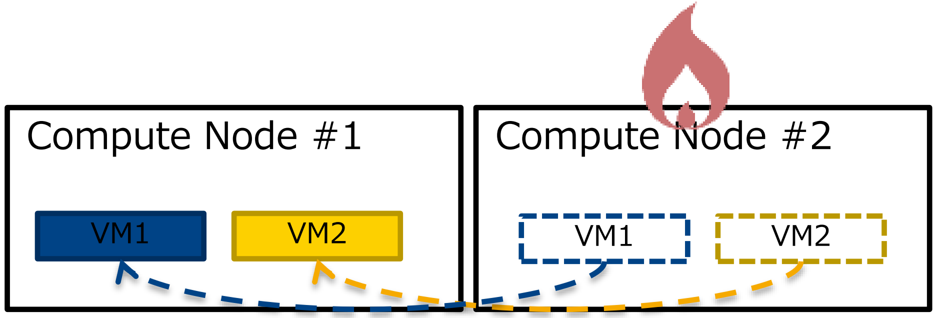
Cattle Model

- Cloud-native IT
- Designed apps for cloud architecture

- Case: If one VM goes down it would not be noticeable to the end-user
- This is the direction for the future

To support pet model: Virtual Machine High Availability

- What is VM-HA
  - Virtual machines on the cloud automatically restart in case of any failure
- Why VM-HA is required in Enterprise Public Cloud
  - From user perspective
    - ✓Minimize impact to traditional IT
  - From Public IaaS provider perspective
    - ✓Keep public IaaS working even if incidents/outage occurs



Challenge: How to implement VM-HA

- OpenStack Community version doesn’t have VM-HA function

- Implement VM-HA to Nova doesn’t match the design concept of Nova/OpenStack
  - Application should be change to cloud native architecture

- If we implement VM-HA to Nova…
  - Maintenance/operational cost increase, so it could create a big obstacle for OpenStack version upgrade

Solution: Masakari, VM-HA in OpenStack

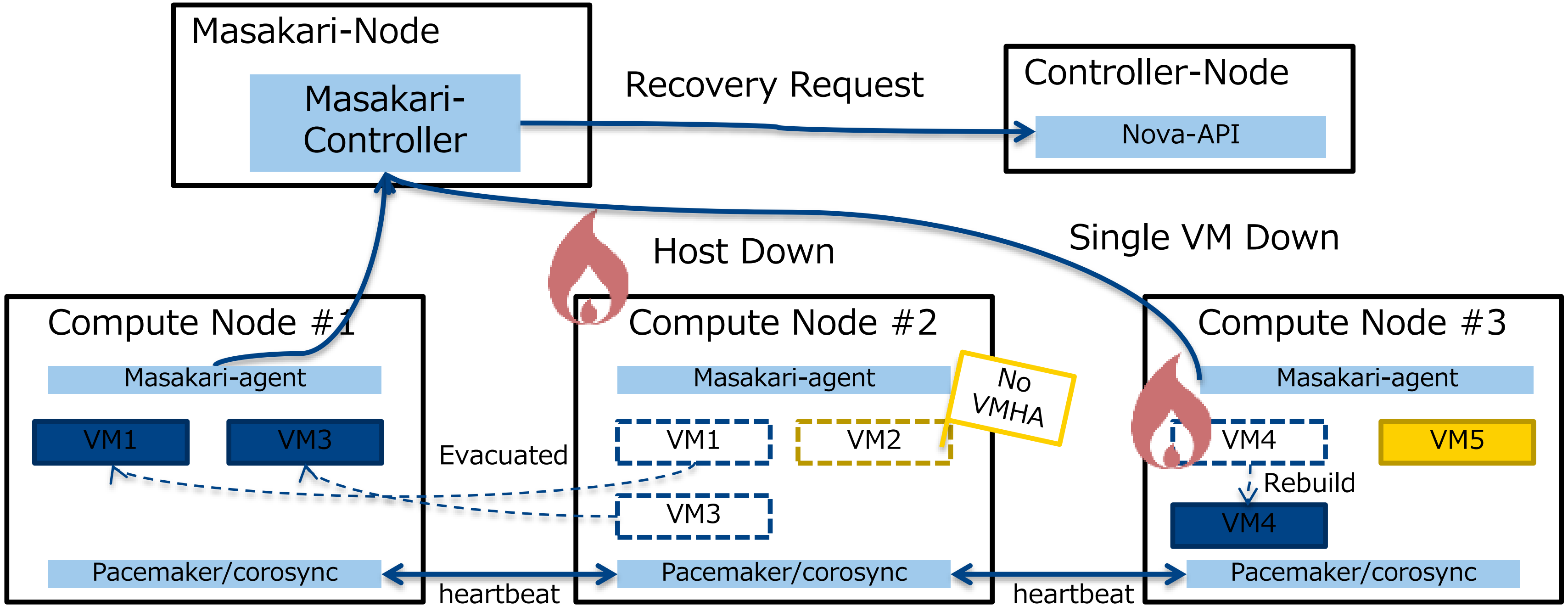
- To realize VM-HA in OpenStack: Masakari
  - Masakari is open source : (https://github.com/ntt-sic/masakari)
  - Extra component / deploy it outside of OpenStack
  - Not need to modify OpenStack's source code

- From service requirement for Pets Model
  - Rescue VM down (VM single down/Host Down)
  - VM recovery within 5mins
  - Work Automatically

- From service requirements for Cattle Model
  - Customer can choose not to use VMHA function provided by Masakari

Solution: Masakari Architecture

- Masakari can rescue a VM affected by the Host Down and Single VM Down incident
  - Masakari(Controller/Agent)・Pacemaker/Corosync



Requirements for Our Service

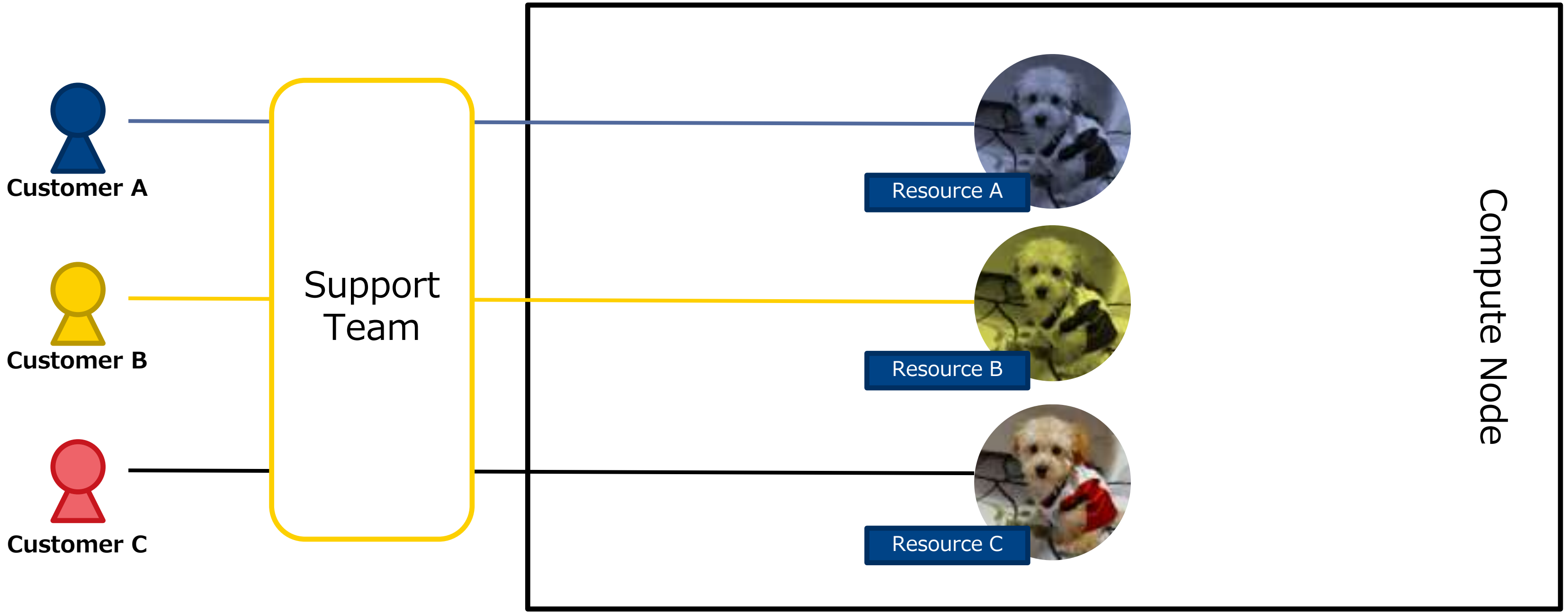
1. **To support both traditional IT and cloud-native IT**
2. Multi-customer/multi-tenant environment

Requirements for Our Service

1. To support both traditional IT and cloud-native IT
2. **Multi-customer/multi-tenant environment**

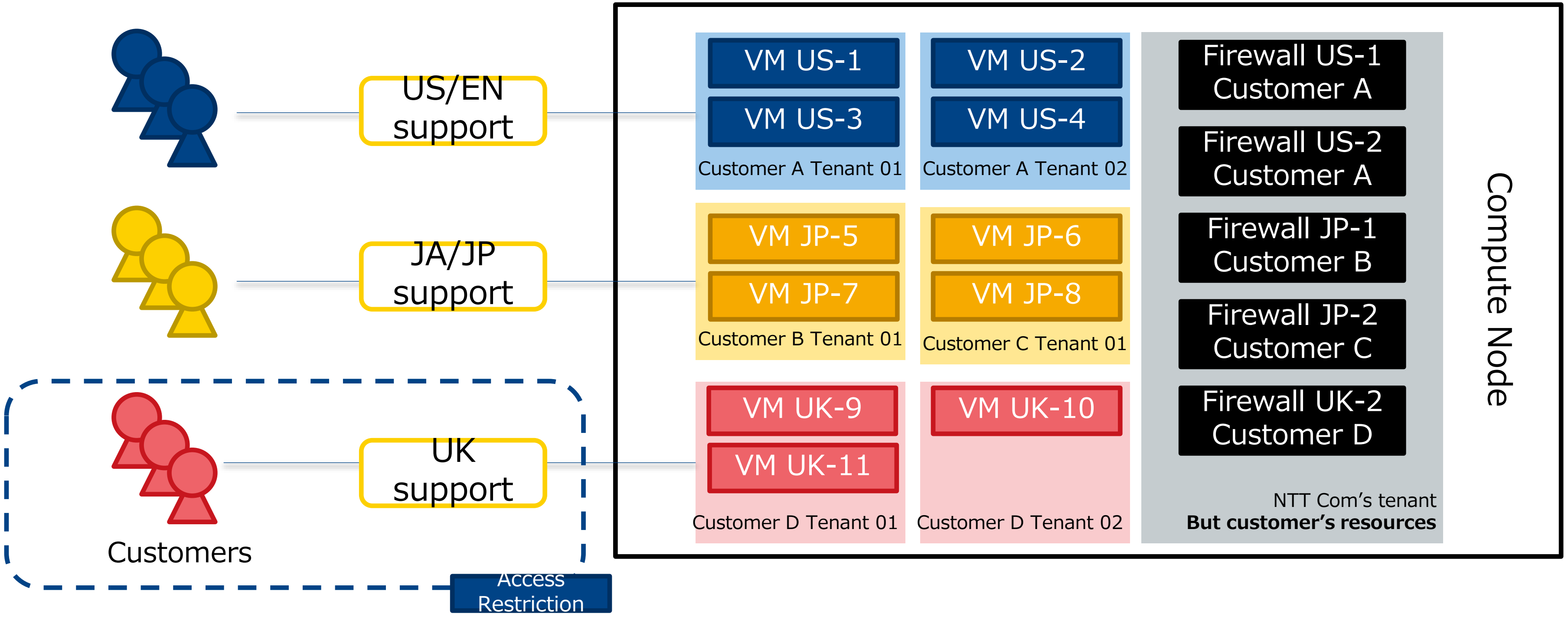
Requirement: Multi-Customer / Multi-Tenant Environment

- In the PET model, when an incident occurs we need to track a lot of information in order to notify the customer



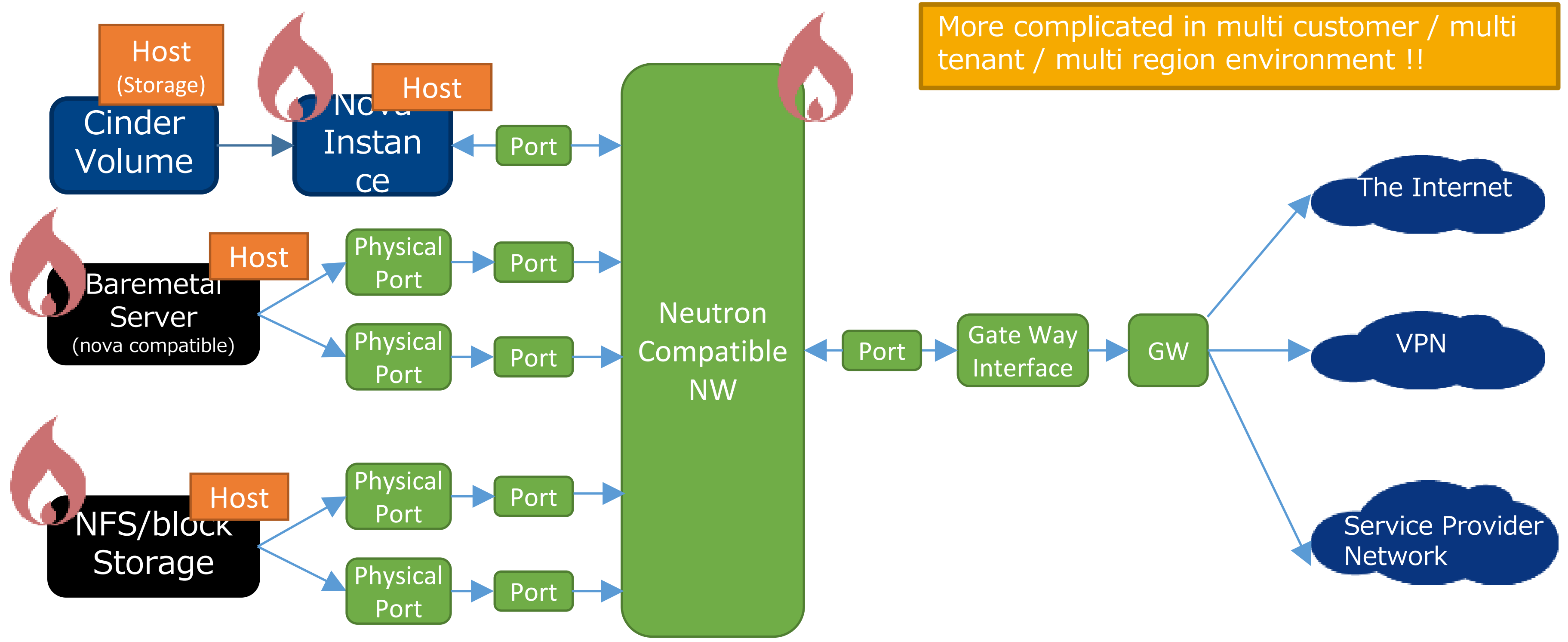
Requirement: Multi-Customer / Multi-Tenant Environment

- Difficult to track in actual environment



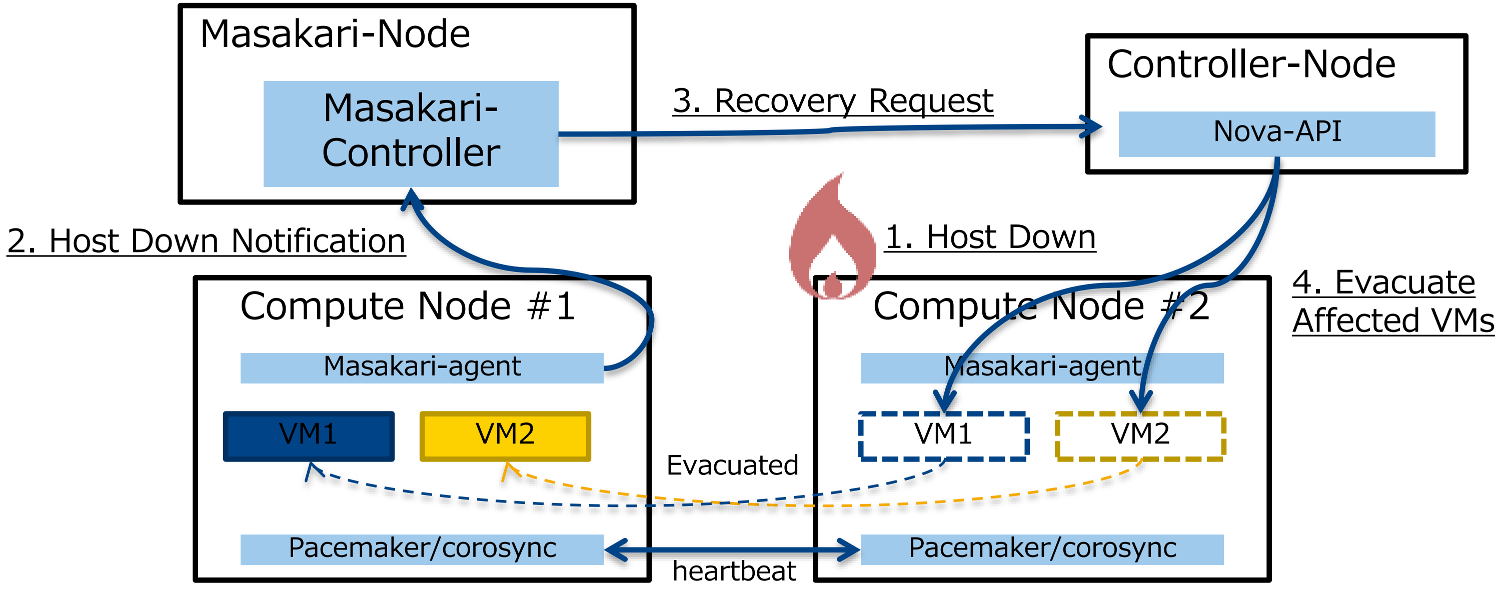
Challenge (1): a lot of Logical/Physical Resources related

- One incident could affects to multiple resources



Challenge (2): VM-HA itself cause missing the VM location

- Sometimes evacuation takes time
- Missing VM location
  - Hard to know which resources has been affected
- Some failures may happen for evacuation itself



Challenge (3): DB Search / Log Tracking

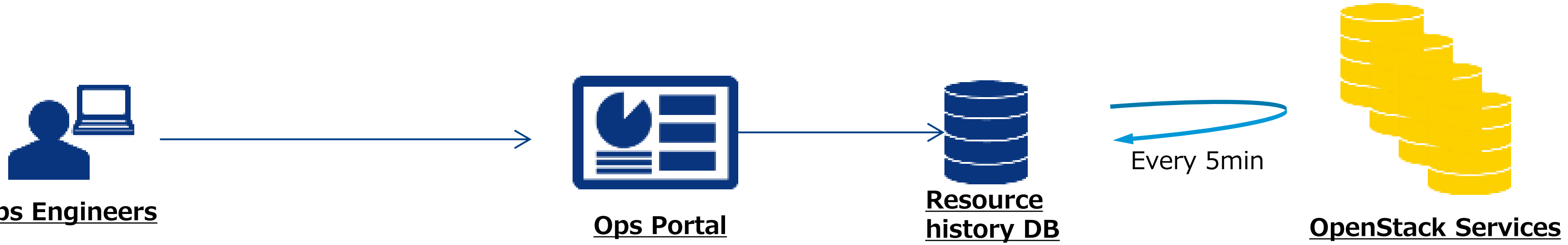
- Searching DB and tracking the relationship of resources is possible BUT:
    - Needs to search across the multiple service DB
    - DB search takes time

- The Masakari log just indicates the log of trigger for VM-HA

- The OpenStack DB shows only the current values and cannot display historical values or statuses

Solution: Operation Portal for Support / Operation Engineers

1. Resource state/location history collection for multiple services
2. Incident Ticket Association with resources information

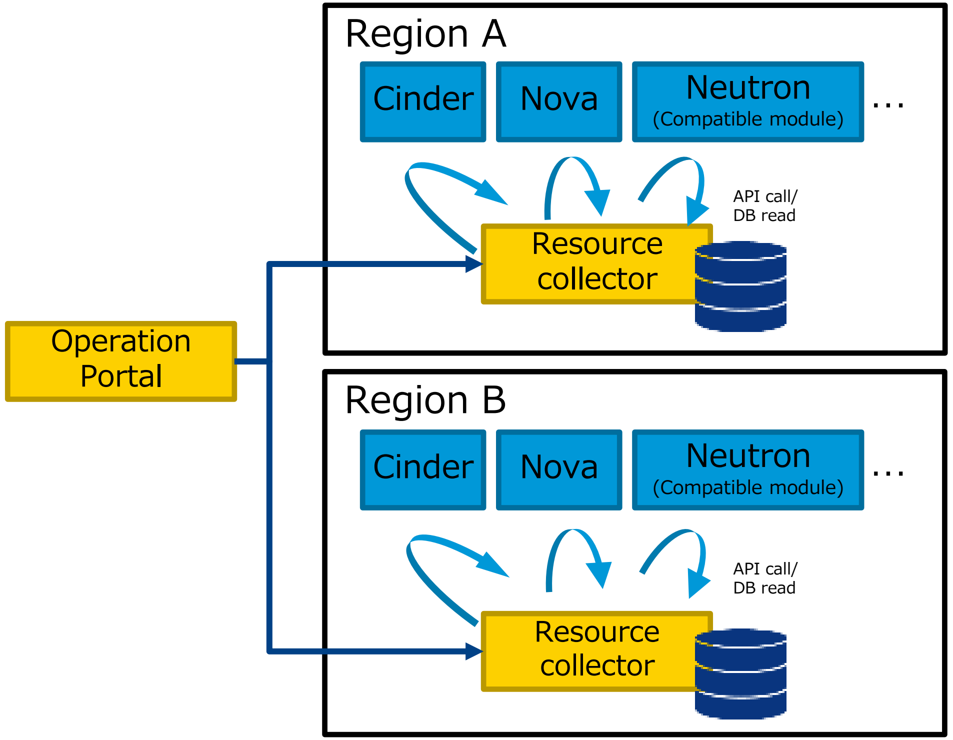


The portal for:
- Check incident ticket
- Check resource relationship mapping
- Check Virtual Resources Location History

Solution (1): Resource state/location history collection

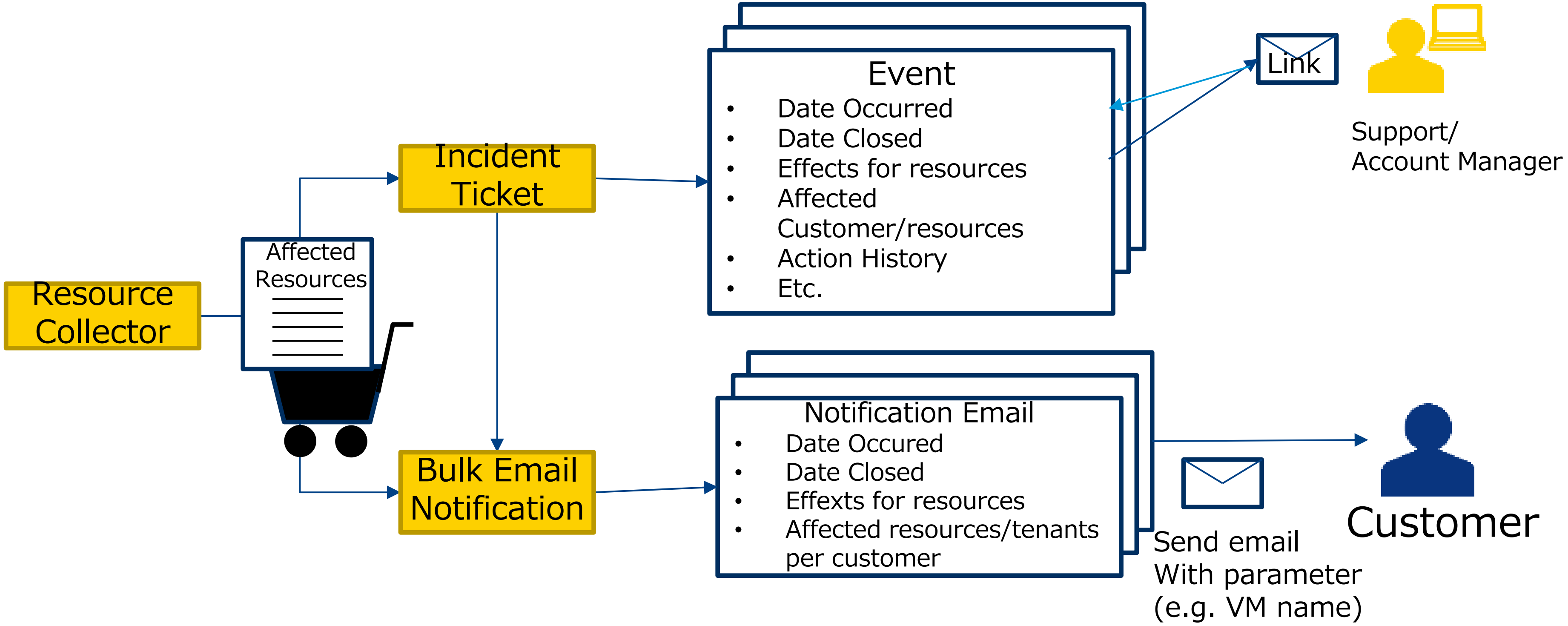
- Collect all historical resource records from OpenStack services
  - Show that resources information for Operators

- Collected Resources
  - Nova (from DB)
    - ✓ instances.*
    - ✓ Instance_metadata
    - ✓ aggregate_metadata
    - ✓ aggregate_hosts
  - Cinder (from DB)
    - ✓ Volumes.*
  - Neutron (from Admin API)
    - ✓ Subnet
    - ✓ Port
    - ✓ IP
    - ✓ etc…



Solution (2): Incident Ticket Association with resources

- All the information is associated with tickets



Actual Use Cases and Demo

VM failover scenario with operation portal

- The host down issue scenario
  1. alert from the monitoring team
  2. operator check which hypervisor gets down and check which VMs are affected
     - Basically VMs are restarted automatically by VM-HA Masakari
  3. send the incident notification
  4. send the recovery notification

Send Notification

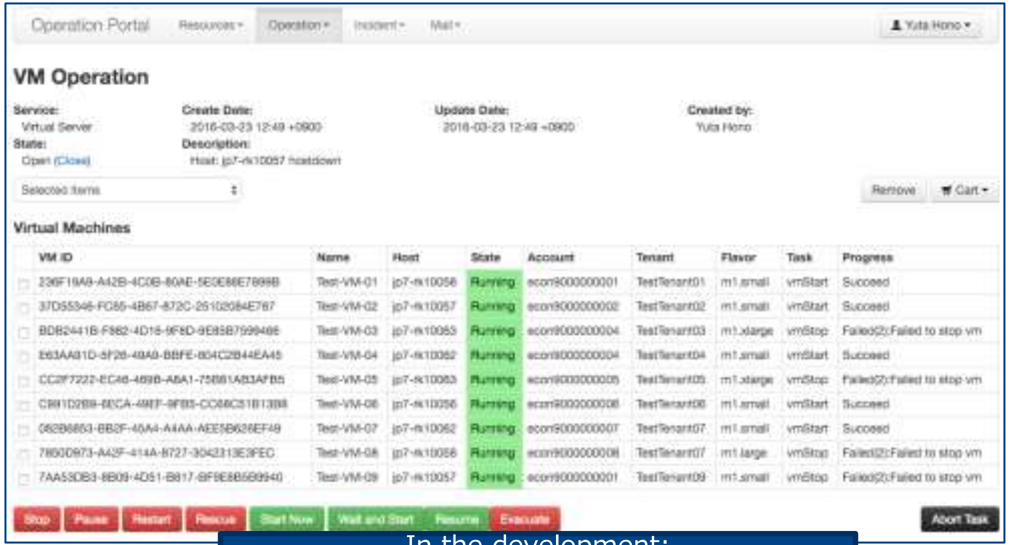
- Use notification template with parameters

```
Thank you for using NTT Communications Enterprise Cloud 2.0 service.
We would like to inform the recovery of following incident.

Tenant
  ID: $tenant.tenantId
  Name: $tenant.tenantName

Affected Your Resources
#if ($vms)
 [Virtual Server]
  $vms
#end
#if ($vfws)
 [Firewall]
  $vfws
#end
```

Velocity Template style statement

Future Enhancement

- Operation Automation / Hand-over to lower Tier Engineer
- Automate incident ticket creation/notification for customer with pre-defined pattern for known-pattern incidents
- Will provide this functionality for our partners also



VM Operation

Service: Virtual Server
State: Open (Close)
Create Date: 2016-03-23 12:49 +0900
Description: Host: jp7-rk10057 hostdown
Update Date: 2016-03-23 12:49 +0900
Created by: Yuta Hono

Virtual Machines

VM ID	Name	Host	State	Account	Tenant	Flavor	Task	Progress
236F19A9-A42B-4C0B-80AE-5E0E86E7898B	Test-VM-01	jp7-rk10056	Running	econ9000000001	TestTenant01	m1.small	vmStart	Succeed
37D55346-FC65-4B67-872C-25102084E787	Test-VM-02	jp7-rk10057	Running	econ9000000002	TestTenant02	m1.small	vmStart	Succeed
BDB2441B-F882-4D16-9F8D-9E85B7599486	Test-VM-03	jp7-rk10063	Running	econ9000000004	TestTenant03	m1.xlarge	vmStop	Failed(2):Failed to stop vm
E63AA01D-5F26-49A9-BBFE-604C2B44EA45	Test-VM-04	jp7-rk10062	Running	econ9000000004	TestTenant04	m1.small	vmStart	Succeed
CC2F7222-EC46-469B-A8A1-75B81AB3AFB5	Test-VM-05	jp7-rk10063	Running	econ9000000005	TestTenant05	m1.xlarge	vmStop	Failed(2):Failed to stop vm
C981D2B9-6ECA-49EF-9FB5-CC66C51B1398	Test-VM-06	jp7-rk10056	Running	econ9000000006	TestTenant06	m1.small	vmStart	Succeed
062B6863-BB2F-45A4-A4AA-AEE5B626EF49	Test-VM-07	jp7-rk10062	Running	econ9000000007	TestTenant07	m1.small	vmStart	Succeed
7860D973-A42F-414A-8727-3042313E3FEC	Test-VM-08	jp7-rk10056	Running	econ9000000008	TestTenant07	m1.large	vmStop	Failed(2):Failed to stop vm
7AA53DB3-8B09-4D51-B817-9F9E8B5B9940	Test-VM-09	jp7-rk10057	Running	econ9000000001	TestTenant09	m1.small	vmStop	Failed(2):Failed to stop vm

In the development:
multiple resources operation from GUI

Conclusion

- Introduced our use case of OpenStack operation under a multi-tenant and multi-customer public cloud environment
  - Achieved quick notification to each customer and recovery VMs affected by incidents with resource history collection / VM-HA Masakari

- Contribution to the OpenStack community
  - NTTCom would keep contributing to the OpenStack community with knowledge from public IaaS operation experiences
    - ✓Feedback / sending patches to community
    - ✓Knowledge sharing with the community in the summit