Elastic Resource Adaptation in the OpenStack Platform

Pedro Martinez-Julia, Ved P. Kafle, Hiroaki Harai

Network Science and Convergence Device Technology Laboratory, Network System Research Institute
National Institute of Information and Communications Technology (NICT)
{pedro, kafle, harai}@nict.go.jp

IEICE Technical Committee on Network Virtualization (NV)
7 March 2018
Outline

• Problem Statement:
  – Motivation and Research Topic
  – Use Case

• Solution:
  – Proposed Approach
  – Architecture Overview
  – Requirement Anticipation
  – Integration with ETSI-NFV-MANO.

• Conclusions & Future Work
Motivation and Research Topic (I)

Trivia:

“High variation in resource demand” vs “Fixed resource allocation”.

[Bar chart showing resource allocation and demand over time ticks.]
Trivia:

“**High variation in resource demand**” vs “**Fixed resource allocation**”.

Answer:

- **Dynamic Environments** → **Elastic Resource Adaptation**
Motivation and Research Topic (III)

Trivia:

"High variation in resource demand" vs "Fixed resource allocation".

Answer:

Elastic Resource Adaptation

Dynamic Environments

Virtual computer and network systems can be dynamically dimensioned to:

- Improve resource utilization.
- Reduce CAPEX.
Motivation and Research Topic (IV)

Trivia:

Virtual computer and network systems can be dynamically dimensioned to:

- Improve resource utilization.
- Reduce CAPEX.

Automated solutions aim to set the optimum dimension for every situation:

- Approach increased system complexity with intelligent and intelligence methods.
- Reduce both OPEX and CAPEX.
Use Case (I)
Use Case (II)

OpenStack Domain 2

OpenStack Domain 3

OpenStack Domain 4

OpenStack Domain 1

HQ Network

Computerized Help Desk
Use Case (III)

OpenStack:
- Facilitates the **construction** of virtual computer and network systems.
  - It is widely used to create production-ready *virtualization environments*.
- Enables the **adaptation** of resources:
  - On-demand instantiation or removal of VMs attached to a service.
- Offers application **interfaces**:
  - Monitoring and resource adaptation.
- Supports **NFV**.
- Its operation will be **enhanced** by the results of our research work.
Use Case (IV)

Closed Link
Overloaded Link
Overloaded Server

OpenStack Domain 2
OpenStack Domain 3
OpenStack Domain 1
OpenStack Domain 4

Computerized Help Desk
HQ Network
Use Case (V)

Underlying virtualization platforms (e.g. OpenStack), require **long time** (~10 s) to be adapted to new requirements:

- **Some client requests could be rejected.**
Most changes in requirements are linked to events from outside the system:

- **User response can be derived from event occurrence.**
- **Required resources can be anticipated to reduce adaptation delay by noticing the events as soon as they occur.**
- **The system can be adapted before the client request burst actually reaches the servers.**
Proposed Approach (II)

Autonomic Resource Control Architecture

- OpenStack Network (D1)
- OpenStack Network (D2)
- OpenStack Network (D3)
- OpenStack Network (D4)

ARCA Engine

System Events
Other Events
Control Actions
Proposed Approach (III)

Collect observations from multiple sources:
- System elements:
  - Underlying controllers (OpenStack), VM monitors, …
- Environment:
  - External event detectors…
Proposed Approach (IV)

Analyze the observations to find out the specific situation of the system:
- Apply administrative policies and control statements to check resource state.
Proposed Approach (V)

Adapt assigned resources:
- Set **resource boundaries** according to found situations.
- Set specific **resource amount** according to estimated demands.
- Issue **actions** to the underlying **infrastructure** (OpenStack controllers).

Controller

ARCA Engine

OpenStack Controller

OpenStack Network (D1)

OpenStack Network (D2)

OpenStack Network (D3)

OpenStack Network (D4)

Controller

Computerized Help Desk

© National Institute of Information and Communications Technology
Proposed Approach (VI)

Delay less than 1 second:
- From external event occurrence to action enforcement.
- Support processing thousands of observations per second.
Proposed Approach (VII)

Challenges, Solutions, and Tools:

- **Too much observations** and volatility:
  - *Input filtering*:
    + Ensure information is not lost and underlying system is not overstressed.

- **Reduce delay**:
  - High performance controller.
  - Anticipate situations (*learning*).

- **Reliability**:
  - Continuous check of policies provided by administrators (statements).
Overview of ARCA (I)

Resources, Controllers, Things  
{Observations: CPU load and sensor readings}  

Collector  

Analyzer  

Decider  

Enforcer  

KB & Reasoner  

Administrator  
{Human}  

Analysis Statements  

Decision Statements  

CEP  

Closed-Loop  

Core  

Resource Controllers
Overview of ARCA (II)

Exploits automation techniques to minimize human involvement:
- Address complex control and management operations.
- Reduce the time required for resource adaptation.
Overview of ARCA (III)

Administrators set operational boundaries for the target system:
- Lower and upper amount of resources that can be assigned.
- Lower and upper load thresholds.

Resources, Controllers, Things
{Observations: CPU load and sensor readings}
Overview of ARCA (IV)

Includes the activities defined by Autonomic Computing (AC):

- Separate micro-services: Collector, Analyzer, Decider, Enforcer
- Closed-loop approach: Check effects of decisions afterwards.

Resources, Controllers, Things
{Observations: CPU load and sensor readings}
Overview of ARCA (V)

Exchanges and knowledge follow a common ontology:
- Encoded in RDF/Turtle and exploiting OWL.
- The ontology can be extended to support new concepts.
- Knowledge is stored in the Fuseki KB, supports SPARQL.

Resources, Controllers, Things
{Observations: CPU load and sensor readings}
Resource Anticipation Strategy

- **Functional and performance target:**
  - *Anticipate* the amount of resources that a controlled system will require before it becomes effective.

- **Involve external event detectors:**
  - Physical: Things (IoT)
  - BigData

- **Learn the event/reaction correlation:**
  - Predict user behavior.
  - Correct mistaken predictions:
    - Improve and optimize learned model...

- **Limit the memory** used by the learning algorithm:
  - Keep only the most relevant vectors.

- **Fast adaptation** to big changes:
  - Discard old vectors when resizing.
Control Flow

- **Two key controlled parameters:**
  - Current Resource Amount (**CRA**).
  - Minimum Resource Amount (**MRA**).

- **Two concurrent sub-routines:**
  - **Anticipation.**
  - Threshold checking and **correction**.

- **Self-assessed learning process:**
  - Correcting learned data when finding mistakes...
Algorithm (I)

1: **procedure** CONTROL(detectors, resources)
2:     mra ← MIN_RESOURCES
3:     cra ← mra
4:     anticipator ← LEARNERCREATE(MIN_RESOURCES, MAX_RESOURCES)
5:     ant_severity, ant_time, ant_peak, ant_peak_rel_time ← 0, 0, 0, 0
6:     severity, pseverity, load, drate ← 0, 0, 1, 0
7:     attl_model ← LEARNERCREATE(MIN_ATTIL, MAX_ATTIL)
8:     **while** TRUE **do**
9:         severity ← COLLATESENSORREADINGS(COLLECT(detectors, SEVERITY))
10:        load ← CALCULATEAVGLOAD(COLLECT(resources, LOAD))
11:       drate ← CALCULATEAVGDROP_RATE(COLLECT(resources, DROP_RATE))
12:      **if** ant_severity ≠ 0 **then**
13:          demand ← cra \* \frac{load + drate}{SERVER_{WORK/QCAP}}
14:          **if** demand > ant_peak **then**
15:              ant_peak ← demand
16:          ant_peak_rel_time ← (NOW − ant_time) \* 1.25
17:       **end if**
18:     **if** NOW − ant_time > LEARNERGET(attl_model, ant_severity) **and** load < LT **then**
19:         LEARNERSET(anticipator, ant_severity, \frac{ant_peak}{HT})
20:         LEARNERSET(attl_model, ant_severity, ant_peak_rel_time)
21:     **if** ant_severity ≠ 0
22:     **if** ant_time ≠ 0
23:        mra ← MIN_RESOURCES
24: **end if**
25: **end if**
Algorithm (II)

26: if severity ≠ 0 and severity ≠ pseverity then
27:     ant_severity ← severity
28:     ant_time ← NOW
29:     ant_peak ← 0
30:     mra ← LEARNERGET(anticipator, severity)
31: end if
32: pseverity ← severity
33: nra ← cra
34: if load > HT or cra < mra then
35:     nra ← MIN(MAX(cra + INC, load * cra / HT), mra), MAX_RESOURCES)
36: end if
37: if load < LT then
38:     nra ← MAX(nra – DEC, mra, MIN_RESOURCES)
39: end if
40: if nra ≠ cra and NOSIDEFFECT(nra) then
41:     cra ← nra
42:     ENFORCE(cra)
43: end if
44: end while
45: end procedure
Alignment With ETSI-NFV-MANO (I)

ARCA-based Engine
{Virtual Infrastructure Manager (VIM)}

Directed by “statements” (policies / rules)

Out of scope

Adapted to “underlying infrastructure providers”, enlarged with external event detectors

© National Institute of Information and Communications Technology
**ARCA** is fitted as the **Virtual Infrastructure Manager (VIM)**:
- Discharges responsibilities from **VNFM** and **NFVO**.
- Improves the **scalability** and **resiliency**...
  ...in case of disconnection from the orchestrator.
- Meets requirements of **Virtual Network Operators (VNOs)**.

**ARCA-based Engine**
{Virtual Infrastructure Manager (VIM)}

**Directed by “statements”**
(policies / rules)

**Out of scope**

Adapted to “underlying infrastructure providers”,
enlarged with external event detectors
Alignment With ETSI-NFV-MANO (III)

The **Nf-Vi** interface (IFA004, IFA019) in **ARCA** has been:
- **Bound to available underlying and overlying interfaces:**
  - Ceilometer/Gnocchi provided by OpenStack.
- **Extended to enable interactions with external elements:**
  - Physical / environmental event (incident) detectors.
  - Big Data: analyzers, data sources, etc.

Adapted to “underlying infrastructure providers”, enlarged with external event detectors.
The Or-Vi interface (IFA005) is provided by:

- The specification of **control/mngmt targets** (statements):
  - Represent the rules and policies that ARCA must enforce.
  - Provided by system administrators and/or external orchestrators.
- **ARCA** will enforce the statements in response to changes in the environment and/or user requirements:
  - Requirements are communicated with additional statements.

**Alignment With ETSI-NFV-MANO (IV)**
Alignment With ETSI-NFV-MANO (V)

The **Vi-Vnfm** interface (IFAO006) is currently out of the scope of **ARCA**:
- Depends on the availability of a proper software (or module) that implements the functions of the VNFM.

Adapted to “underlying infrastructure providers”, enlarged with external event detectors.
Conclusions & Future Work

• Designed **ARCA**:  
  – To provide functions of the Virtual Infrastructure Manager (**VIM**) of **NFV-MANO**.  
  – Extended **VIM interfaces** to meet requirements of the **real world**:  
    • Sport events, TV shows, emergency scenarios...  
  – Achieved good performance within an OpenStack-based deployment:  
    • Detailed **overlying** and **underlying** infrastructures.  
    • Reproduction of **production-like environments** to ensure **transferable research results**.

• SDN/NFV and OpenStack **stakeholders** will **benefit** from ARCA features:  
  – **Efficient** use of **resources**:  
    • Further reduce CAPEX and OPEX:  
  – Benefit to both **infrastructure providers** and **consumers**.

• Next steps:  
  – Keep **reducing** ARCA **response time**.  
  – Increase **complexity** of the **validation scenario**:  
    • Mix clients and servants in the same domains.  
  – Align **ARCA-based VNC** to **additional requirements** from **NFV/SDN**.
Thanks for Your Attention
Q & A