HORIZON2020 FRAMEWORK PROGRAMME

TOPIC EUK-03-2016

"Federated Cloud resource brokerage for mobile cloud services"

D3.4
BASMATI Service Monitoring Framework

Project acronym: BASMATI

Project full title: *Cloud Brokerage Across Borders for Mobile Users and Applications*

Contract no.: 723131

Workpackage:	4	Dynamic Brokerage and Federation: Software Prototype	
Editor:	Cheolyong Jo		INNO
Author(s):	Cheolyong Jo		INNO
Authorized by	Young Woo Jung Konstantinos Tserpe		ETRI ICCS
Doc Ref:	3.4		
Reviewer	K. Tserpes		ICCS
Dissemination Level	Public		

Document History

Version	Date	Changes	Author/Affiliation
v.1.0	18/04/2018	Version for review	Cheolyong Jo, INNO
v.1.0_KT	24/04/2018	Commented version	K. Tserpes, ICCS
v.1.1	27/04/2018	Updates to the Comment	Cheolyong Jo, INNO

BASMATI Glossary

Term/Acronym	Definition
Mobile cloud services	Online services offered by cloud resources to support mobile apps. The backend of the mobile apps.
CP	Cloud Provider. The actor that provides the cloud infrastructure/resources, such as VMs
CSP	Cloud Service Provider. The actor that provides cloud services on top of a rent infrastructure from a CP
Cloudlet	Limited capacity infrastructures with virtualization capabilities, often used to support a limited amount of users or perform a limited set of operations on behalf of the central cloud infrastructure that hosts the complete application
Edge resources	Resources aimed to operate specialized functionality, located at the "edge" of the network infrastructure, thus, closer to the end users. Examples are (clusters of) RaspberryPis or cloudlets
BUDaMaF	BASMATI Unified Data Management Framework
KE	Knowledge Extractor
DM	Decision Maker
RB	Resource Broker
MVD	Mobile Virtual Desktop
DASFEST	An 3-day long music festival taking place in Karlsruhe, Germany every July
ACE	Amenesik Cloud Engine. The cloud service deployment tool through which actual federation is achieved
BEAM	BASMATI Enhanced Application Model. An extension of the TOSCA specification
ASP	Application Service Provider. A Federation user that rents resource services in order to provide an Application services to End-users
Brokering	The matchmaking support provided by BASMATI platform to decide about the best cloud resources to exploit for the execution of the back-end of BASMATI applications. This activity regards the placement of the services or data on computational resources and storages belonging to the cloud data centre and the cloudlets within the federation.
End user	A user who benefits the various application and infrastructure services provided by the Cloud. Within BASMATI, the most typical example is exploiting the Cloud federation via a mobile device (possibly a laptop) using specialized apps or a web browser.
Offloading	The ability of BASMATI platform supporting the runtime placement of the components composing the front-end of BASMATI applications on edge resources available nearby the end user. This activity takes place both when edge and mobiles exchange one each other their own workload or when such devices transfer some workload to the clouds or cloudlets. In BASMATI we often distinguish Front-end offloading, related to the mobile part of application, from Back-end offloading, concerning the server side of applications. The latter roughly translates to the known concept of Cloudbursting.
QoE	Quality of experience. It is a measure of a customer's experiences with a

	service. It may be related to some aspects of the QoS and QoP, but can also take into account other metrics.
Service handover	Service handover refers to the activity of transferring an active service between two computational resources (e.g. Cloudlets) with minimal or no disruption on the availability of the service. Ideally, service handover is transparent with respect to the user.
Situational Awareness	The ability of the BASMATI platform to recognise the “situation” characterising the actual combined status of users, applications and resources, aimed at achieving an effective and efficient management of applications and resources.

Executive Summary

This framework provides the monitoring functionality of the BASMATI service. Through the framework, you can check the status of the CSP, VM, and application of the BASMATI service. It is also possible to achieve VM monitoring. Finally, it provides APIs for interworking with other compo.

Table of Contents

1 Introduction

This framework provides monitoring of the BASMATI service. Through this framework, you can check the status of CSP, VM, and application of BASMATI service. And VM monitoring is possible. It also provides related APIs for interworking with other projects.

1.1 Relationship to Other Deliverables

This framework is based on the federation monitoring as reported in the BASMATI deliverable D3.3 “Situational Knowledge Acquisition and Inter-Cloud Service Monitoring”.

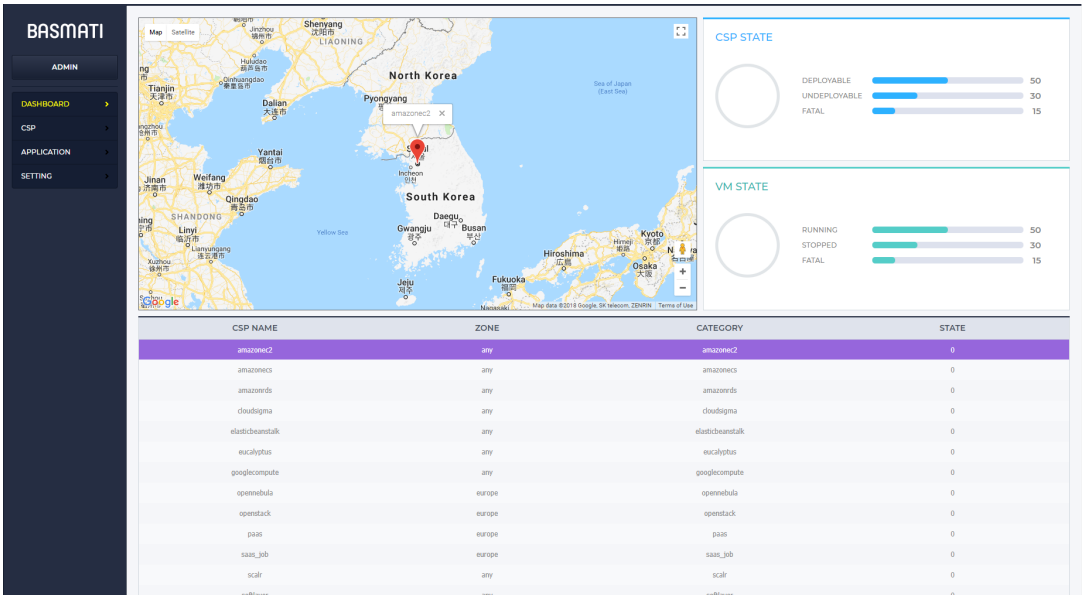
1.2 Scope of the framework

This framework includes:

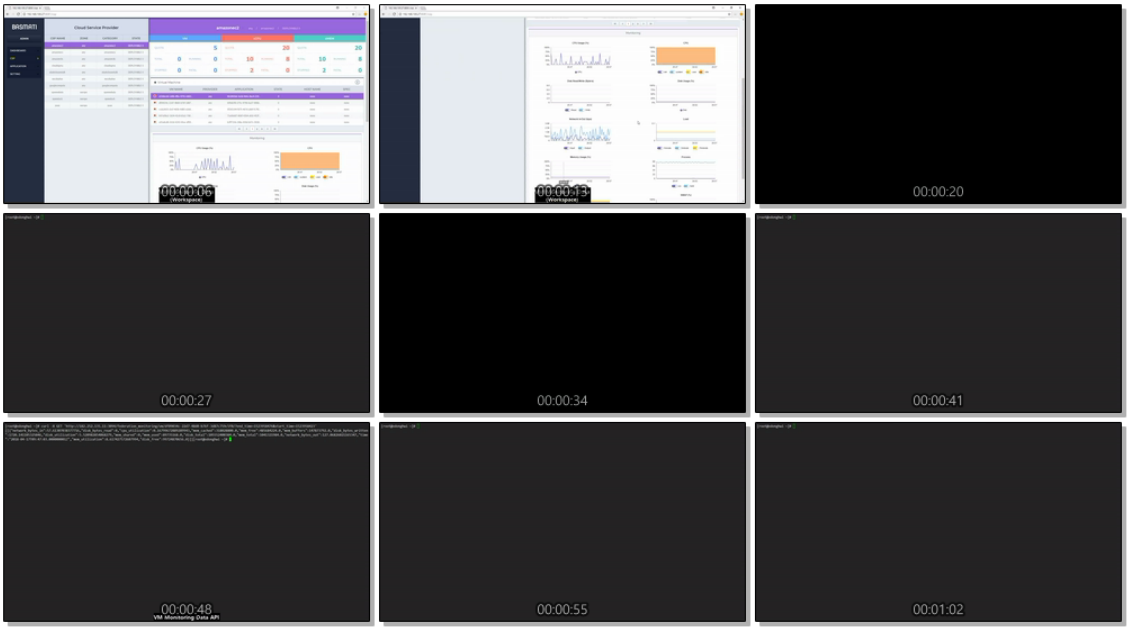
- CSP Status
- VM Status
- Application Status
- Relationship between CSP and VM
- Relationship between VM and Application
- VM Monitoring, VM List API
- VM Monitoring API
- VM Alarm API

2 Framework Demonstration

Framework Demonstration Web Page



Framework Demonstration Video(Snapshot)



2.1 Location

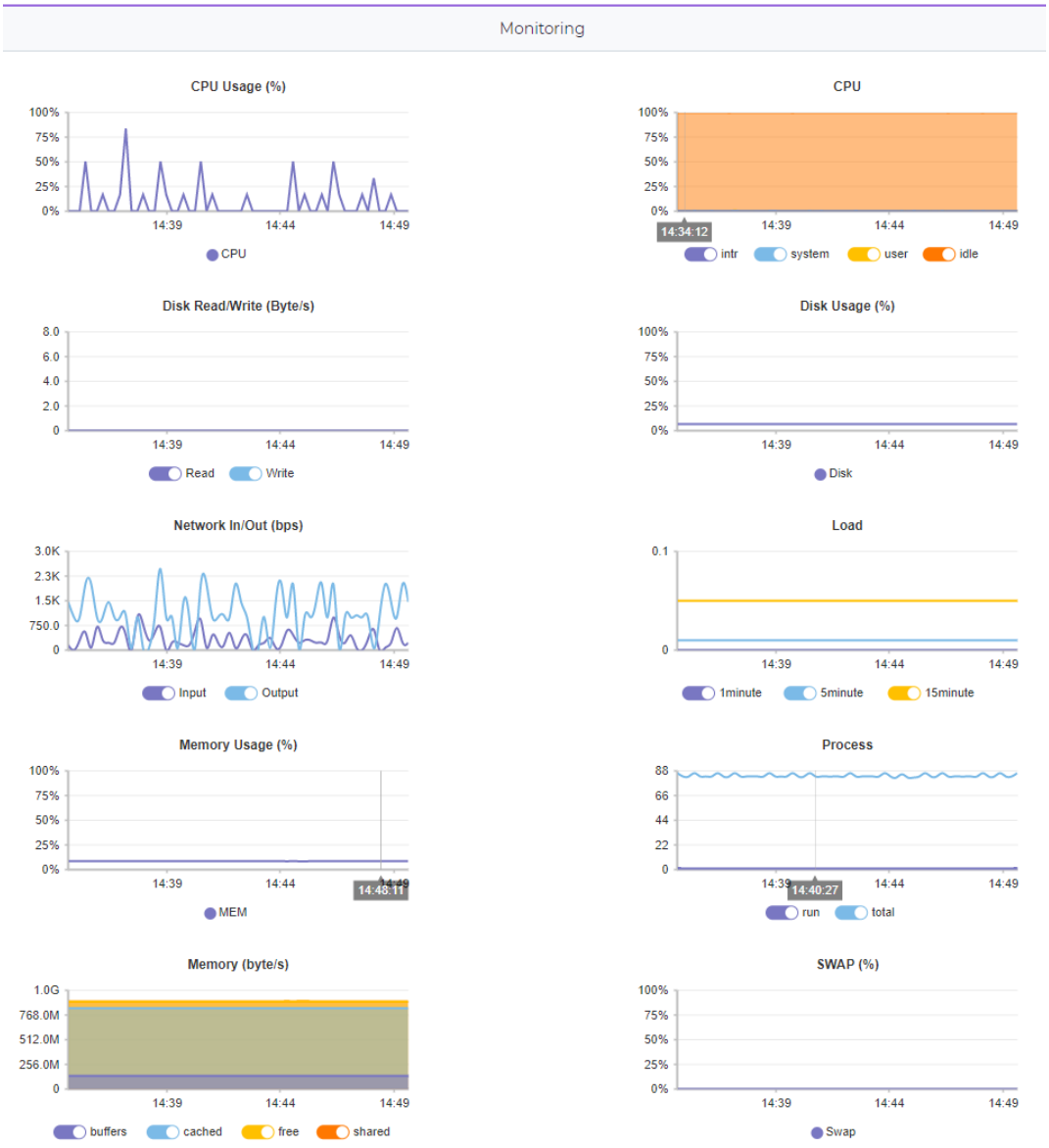
Demonstration Video Youtube URL : https://youtu.be/LZOU9nhL228

Demonstration Web Page URL : http://182.252.131.51:8081/

2.2 Description

This video is a demonstration of the BASMATI Service Monitoring Framework.

The first part is the VM monitoring demonstration and displays the monitoring data of each VM. Below is the VM monitoring graph. More information about monitoring metrics is provided in the VM Monitoring API.



The Second part is VM List API demonstration. You can request a list of VMs in the BASMATI service. The API is provided through the Federation Monitoring module And the data collected from CMP(ACE). The API information is as follows.

Request Method	GET
Request Description	Request VM List
Request Parameter	
-	
Request Example	
curl -X GET http://182.252.135.11:3094/federation_monitoring/vm/	
Response Content	
vm_id : UUID url : String provider : String image : UUID application : UUID price : String state : String	ip : String os : String cpu : String mem : String disk : String collector_id : String deleted : String
Response Example	


```
{
    "vm_id": "0d4608e6-030a-49b7-8a79-ed5dd7701493",
    "url": "",
    "provider": "any",
    "image": "0881a61d-7684-4f85-a0b8-bdac01c9efd3",
    "application": "3b2a73ee-169f-4478-b4d2-342d7f53c516",
    "price": "",
    "state": "0",
    "hostname": "none",
    "ip": "none",
    "os": "none",
    "cpu": "none",
    "mem": "none",
    "disk": "none",
    "collector_id": "none",
    "deleted": "no"
  }
```

The Third part is VM Monitoring API demonstration. You can request monitoring data of each VM in the BASMATI service. The API is provided through the Federation Monitoring module. The API information is as follows.

<table>
<tr><td>Request Method</td><td>GET</td></tr>
<tr><td>Request Description</td><td>Request VM Monitoring Data</td></tr>
<tr><td colspan="2">Request Parameter</td></tr>
<tr><td colspan="2">vm_id : UUID
start_time : UNIX time
end_time : UNIX time</td></tr>
<tr><td colspan="2">Request Example</td></tr>
<tr><td colspan="2">curl -X GET
http://182.252.135.11:3094/federation_monitoring/vm/{vm_id}/?start_time={start_time}&end_time={end_time}

curl -X GET
http://182.252.135.11:3094/federation_monitoring/vm/df09034c-22d7-48d0-b76f-3d87c759c5f0/?start_time=1523958423&end_time=1523958`476</td></tr>
<tr><td colspan="2">Response Content</td></tr>
<tr><td>cpu_utilization : percentage
mem_utilization : percentage
mem_total : long
mem_free : long
mem_used : long
mem_cached : long
mem_buffers : long
mem_shared : long</td><td>disk_utilization : percentage
disk_total : long
disk free : long
disk_bytes_read : long
disk_bytes_written : long
network_bytes_in : long
network_bytes_out : long</td></tr>
<tr><td colspan="2">Response Example</td></tr>
</table>

```
{
        "network_bytes_in": 57.61307038377716,
        "disk_bytes_read": 0,
        "cpu_utilization": 0.16799672889289943,
        "mem_cached": 318028800.0,
        "mem_free": 485684224.0,
        "mem_buffers": 147873792.0,
        "disk_bytes_written": 1720.141105325046,
        "disk_utilization": 5.520982854068279,
        "mem_shared": 0,
        "mem_used": 89735168.0,
        "disk_total": 105552400384.0,
        "mem_total": 1041321984.0,
        "network_bytes_out": 127.06826021165347,
        "time": "2018-04-17T09:47:03.000000001Z",
        "mem_utilization": 8.617427572607994,
        "disk_free": 99724870656.0
    }
```


The fourth part is Alarm API demonstration. You can registry alarm information of each VM in the BASMATI service. When the alarm conditions are met, the alarm is called. The alarm call part has not been developed yet. The API is provided through the Federation Monitoring module. The API information is as follows.

Request Method	Post
Request Description	Registry VM Alarm
Request Parameter	
vm_id : UUID name : String alarm_type : String operator : String check_cycle : Integer check_count : Integer value : Integer	
Request Example	
curl -H "Content-Type: application/json" -X POST -d '{"instance_id": "{vm_id}"name": "{name}", "alarm_type": "{alarm_type}", "operator": "{operator}", "check_cycle": {check_cycle}, "check_count": {check_count}, "value": {value}}' http://182.252.135.11:3094/federation_monitoring/alarm/ curl -H "Content-Type: application/json" -X POST -d	

'{"instance_id": "df09034c-22d7-48d0-b76f-3d87c759c5f0", "name": "cpu alarm3", "alarm_type": "cpu_utilization", "operator": "gt", "check_cycle": 4, "check_count": 5, "value": 20}' http://182.252.135.11:3094/federation_monitoring/alarm/
Response Content
Id : UUID vm_id : UUID name : String alarm_type : String operator : String check_cycle : Integer check_count : Integer value : Integer
Response Example

{
"id":4,
"alarm_id":"d88d195e-4157-4cea-983f-b0a072e3e9d0",
"instance_id":"df09034c-22d7-48d0-b76f-3d87c759c5f0",
"name":"cpu alarm3",
"alarm_type":"cpu_utilization",
"operator":"gt",
"check_cycle":4,
"check_count":5,
"value":20,
"description":null
}

3 Prototype Installation Guide

We also provide the source code and installation instruction for readers to try the software.

3.1 Authentications

The source code is not yet registered in the repository.

3.2 Source Code Location

The source code is not yet registered in the repository.

3.3 Requirements

These are the list of initial requirements in order to deploy the software:

- CentOS 7
- Java OpenJDK 1.8.0

3.4 Installation Guide

1. Git pull source code
2. #service basmati-workspace start

The source code of the BASMATI Federation Monitoring can be downloaded from the Basmati Git Lab repository, located at the following URI:

git@basmati.amenesik.com:code/platform/modules/federation_monitoring.git

using the following credentials:

- username: guest
- password: basmati_review7012

4 Conclusions

We have developed a framework for monitoring and managing different CSPs and VMs. We have also developed an API for interworking with other modules.

In the future, more CSPs will be serviced, and cloud users will be using various CSPs together. In this case, services to manage and monitor the various CSPs are essential. The framework we have developed may be one approach.