Cloud based Health-Care Platform

Ki-Hyun Kim, In-Yong Jung, Byong John Han and Chang-sung Jeong
Department of Electrical Engineering, Korea University
Seoul, South Korea
[e-mail: cocoball@korea.ac.kr]
*Corresponding author: Chang-sung Jeong

Abstract

Existing health-care platforms for patients' health care data collection require a great deal of workload to collect, input and analyze the medical data. These platforms are not able to support the requirements of massive medical data storage, management and analysis. To solve the problem, we proposes Cloud based Health-Care Platform. It supports massive heath-care data management and computation intensive work such as 3D volume rendering using hybrid resource management method. Also, it provides a scalable high-performance medical collaboration environment by supporting auto-scaling engine. We shows the results of implementing the auto-scaling engine in our platform where dynamic resource allocation. We demonstrate the performance of our platform in an experimental evaluation.

***Keywords*:** Health-care, Cloud computing, Medical data

1. Introduction

Nowadays, health-care platform becomes more complicated and grows bigger by expanding health care area. Modern health care system for patients' data collection support RDBMSs-based framework. It is not able to support the requirements of massive health-care data storage, management and analysis. Also, current medical image processing applications require management of huge amounts of data and executive high-performance computing.

In this paper, we propose Cloud based Health-Care Platform which supports automated medical collaboration environment such as data management, and medical image processing. It provides dynamic resource allocation to provide system extendibility and hybrid job execution environment for computational intensive tasks such as image processing.

2. System Architecture

2.1 Architecture

Cloud based Health-care Platform composed of four components: Medical Collaboration Space Manager (MCM), Medical Data Repository (MDR), Medical Image Processing Engine (MPE), and Cloud Manager (CM).

2.2 Medical Collaboration Space Manager (MCM)

MCM provides a web user interface for medical collaboration space management service. **Fig. 1** shows the detailed architecture of MCM. Users are mainly composed of medical collaboration workers. MCM hosts a group of workspaces for medical collaboration between massive user groups. They use the system for data sharing, integration and analysis. Users through a

This research was supported by the MSIP (Ministry of Science, ICT & Future Planning), Korea, under the ITRC (Information Technology Research Center), support program (NIPA-2014-H0301-14-1001) supervised by the NIPA (Nation IT Industry Promotion Agency), and Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the MSIP (NRF-2013043678).

web-based MCM to interact with our platform. MCM provides three interfaces for user management, medical collaboration management, and medical collaboration space management.

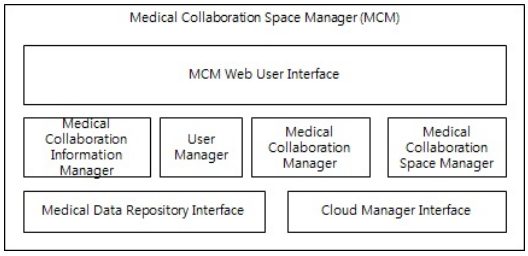

Fig. 1. Architecture of MCM

2.3 Medical Data Repository (MDR)

MDR provides distributed health-care data storage and automated health-care service. **Fig. 2** shows the architecture of MDR. It is composed of 3 working components (Medical Collaboration Work Service, Medical Data Manager and Medical Data Manipulation Service). Medical Collaboration Work Service of MDR provides medical data centric collaboration. Medical Data Manager is connected with HDFS (Hadoop Distributed File System) [1]. HDFS Manager provides scalable file system. When Auto-Scaling Engine of CM detects lack of remaining storage of MDR, it adds new data node automatically.

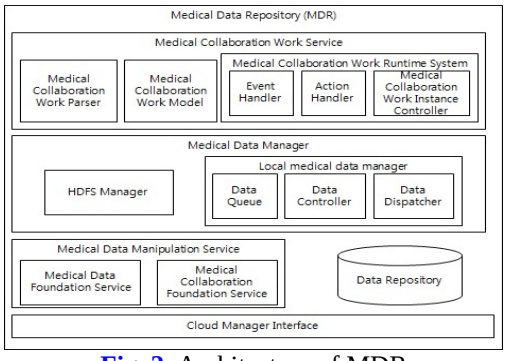

Fig. 2. Architecture of MDR

2.3 Medical Image Processing Engine (MPE)

Our platform can provide computation intensive multimedia applications which support medical collaboration in professional area by using MPE. It is a management component for providing medical image processing service. **Fig. 3** shows the architecture of MPE. MPE consists of two componentsL MPE Interface, and Medical Image Processing Service. When a MRI volume data is sent, Job Receiver receives and executes Medical Image Processing Service such as 3D volume rendering. After image processing task is done, Result Sender sends the result to MDR for health care service. Unlike other cloud based platforms, out platform provides hybrid job execution environment by combining both VM and physical resource. Generally, processing speed of physical machine is much faster than VM. Futhermore, common VM has functional limitation compared with physical machine. Our platform supports fast shear warp 3D volume rendering service [2]. However, our GPU based 3D volume rendering service is not supported in VM. When our platform receives 3D volume rendering tasks, the input data of volume rendering send to physical resource for execute rendering tasks using GPU. **Fig. 4** shows the result image of our 3D voule rendering service.

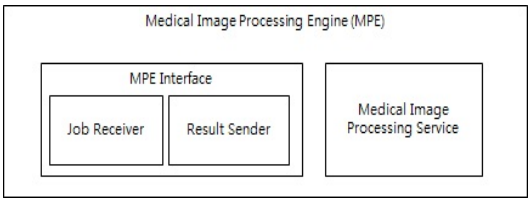

Fig. 3. Architecture of MPE

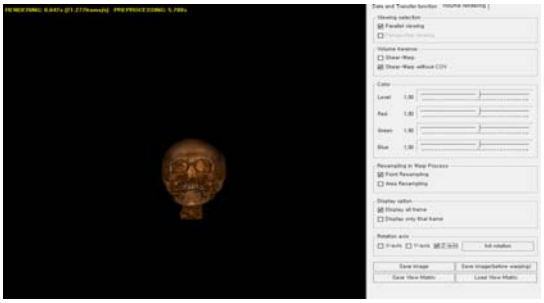

Fig. 4. Result image of 3D volume rendering

3. Cloud Manager

CM is a resource management component which provides all computational resources used in medical collaboration platform by using cloud infrastructure system. It is composed of four components: Auto-Scaling Engine, Resource Prediction Manager, Resource Manager and

Resource Agent Manager. The architecture of CM is shown in **Fig. 5**.

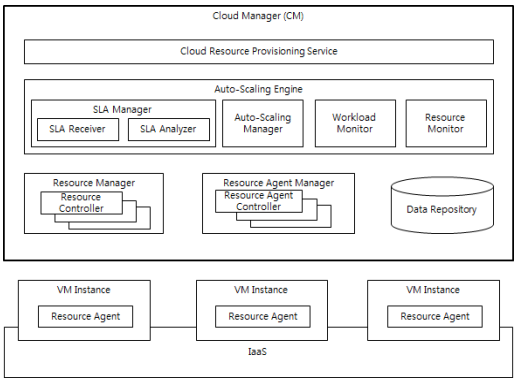

Fig. 5. Architecture of Cloud Manager

To manage cloud resources dynamically based on response time and hardware related resource requirements, we developed Auto-Scaling Engine. Auto-Scaling Engine includes four components. They are SLA Manager, Auto-Scaling Manager, Workload Manager and Resource Manager. SLA Manager receives and analyzes the user's SLA data which includes the auto-scaling factor. Auto-Scaling Manager collects the information from Workload Monitor and Resource Monitor and detects violations of auto-scaling factor auch as response time or hardware requirements (CPU, Memory, Storage etc). Workload Monitor monitors the logs of the workload. It reads the workload logs in real time and calulates the average response time for each VM over intervals of 60 seconds. Resource Monitor monitors the hardware related resource information. **Fig. 6** shows the pseudo code of Auto-Scaling Manager.

```
set SLA_cu = 30.0 {Maximum CPU utilization (%) allowed in SLA}
set SLA_mu = 50.0 {Maximum Mem utilization (%) allowed in SLA}
set isScaling = false
while true do
  ResourceMonitor.ReadResourceInfo(instanceID)
  for each node in ACCP do
    ResourceMonitor.CPUutilization(node)
    ResourceMonitor.MemUtilization(node)
  end for
  if Avg_cu of any node > SLA_cu and isScaling == false then
    isScaling = true
    instanceId = ResourceManager.instantiateVirtualMachine()
    vmip = ResourceManager.getVMIP(instanceId)
    AutoScalingManager.addVMtoACCP(vmip)
    isScaling = false
  else if Avg_mu of any node > SLA_mu and isScaling == false then
    isScaling = true
    instanceId = ResourceManager.instantiateVirtualMachine()
    vmip = ResourceManager.getVMIP(instanceId)
    AutoScalingManager.addVMtoACCP(vmip)
    isScaling = false
  end if
end while
```

Fig. 6. Pseudo code of Auto-Scaling Manager

Whenever Resource Monitor detects the resource information of any VM exceeds the required resource information, it invokes the initalVirtualMachine method of Resource Manager with the required parameters and obtains a new VM instance ID.

4. Conclusions

In this paper, we have described a Cloud based Health-Care Platform that real-time monitors the hardware requirements and web response time of VM hosted on our platform for health-care service and dynamically scales up and down the VM resources to satisfy a platform administrator defined SLA. Dynamic resource management using Auto-Scaling Engine in clouds would process computation intensive work such as 3D volume rendering using hybrid resource management method.

References

[1] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler : The Hadoop Distributed File System, IEEE, 2010.

[2] Kiyoung Choi, Sungup Jo, Hwamin Lee, Changsung Jeong: CPU-based speed acceleration techniques for shear warp volume rendering. Multimed Tools Appl 64, pp. 309-329, 2013.