

San José State University

College of Engineering

Department of Computer Engineering

CMPE 283-01, Virtualization Technology

Fall 2014

**Group Project – 2: LARGE SCALE STATISTICS & ANALYSIS**

Submitted to

Prof. Simon Shim

Submitted By:

Poornima Peddinti(009179066)

Rajas Hegiste(010006438)

Shreya Subhlok(009303606)

Vaibhav Tupe(009606766)

December 7, 2014

Table of Contents

[1. Introduction 3](#_Toc405759083)

[1.1. Goals 3](#_Toc405759084)

[1.2. Objectives 3](#_Toc405759085)

[1.3 Needs 3](#_Toc405759086)

[2. Background 4](#_Toc405759087)

[3. Requirements 4](#_Toc405759088)

[3.1. Functional Requirements 4](#_Toc405759089)

[3.2. Non Functional Requirements 4](#_Toc405759090)

[4. Design 4](#_Toc405759091)

[4.1.Algorithm 5](#_Toc405759092)

[4.2. Architecture 6](#_Toc405759093)

[4.3. Key WorkFlows/Components 6](#_Toc405759094)

[5. Implementation 7](#_Toc405759095)

[5.1. Environment 7](#_Toc405759096)

[5.2. Tools and Technologies 7](#_Toc405759097)

[5.2.1. Application  Performance Management 8](#_Toc405759098)

[5.3. Screenshots 8](#_Toc405759099)

[6. Assumptions 10](#_Toc405759100)

[7. Limitations 10](#_Toc405759101)

[8. Future Work 11](#_Toc405759102)

[9. Individual Contribution 11](#_Toc405759103)

[10. Installation and Execution manual 12](#_Toc405759104)

[12. Testing 15](#_Toc405759105)

[12.1. Unit Testing 15](#_Toc405759106)

[12.2. Integration Testing 15](#_Toc405759107)

[13. Screenshots with annotations 16](#_Toc405759108)

[14. Conclusion 22](#_Toc405759109)

[14.1. Lessons Learned 22](#_Toc405759110)

[14.2. Challenges 22](#_Toc405759111)

[15. References 22](#_Toc405759112)

# 1. Introduction

Virtualization is basically creating an unreal version of something like a server, network, operating system etc. It is a very efficient technique to make the complete use of all the resources to its potential. Virtualization when used properly can be very useful to reduce cost of IT infrastructure. It has different properties like efficiency, scalability, load balancing and easy portability. Many applications are deployed using virtualization without any issues or any threats. These applications run efficiently without affecting their performance.

The project which we(Team04) have done is basically developed a Distributed Resource Scheduler and a Distributed Power Management function and along with that we have gathered large scale statistics and performed analysis in a scalable environment on this statistics.

## 1.1. Goals

* Getting used to the virtualization environment, managing, load balancing and testing VM.
* Gathering of a large scale data from all the Virtual Machines.
* Performing simple analysis on the data that is gathered.
* Representation of data in a chart form using google charts.
* Implementing DRS and DPM algorithms.

## 1.2. Objectives

**Part-1**

* Implementing DRS and DPM to manage the Virtual Machines on the vHost according to the requirement.

**Part-2**

* Monitor Virtual Machine and vHosts
* Gathering the stats of VM and vHosts in logs
* Perform analytics on the stats that are gathered and take the necessary steps.
* Collect the data and store it using various tools.
* Stored data is used to build charts in google chart showing various charts of the stats.

## 1.3 Needs

* To manage the data that is gathered from the Virtual Machine and vHost.
* To perform analytics on them to implement DRS and DPM.
* To perform necessary actions from the chart plotted from the graph.

# 2. Background

Virtualization is one of the most uprising techniques that are out there in the market. It is becoming base foundation of computation for all small scale as well as large scale industries. Virtualization has taken the concept of Resource Utilization to a whole another level. When virtualization did not exist there were not enough ways to make complete use of available resources. A single Operating System was allowed on a single machine which would make it very difficult for a complete use of all the available resources. But once there was a complete usage of such tools there it would be difficult for to get an extension every time such an error occurred . Expanding the resource would prove as a temporary solution.

**Part-1**

Managing the Resources using the DRS and DPM.

**Part-2**

The task of gathering the statistics of all the VM and Hosts is very useful as it gives the idea of what is happening in the resource pool and necessary actions can be taken as and when required. This would be a much better solution to act dynamically against such problems or issues faced even in future.

# 3. Requirements

## 3.1. Functional Requirements

 The functional requirements are the following:

* Jar file is used to extract log from the VM.
* Agent process sitting at the logstash to collect and send monitoring data to the MongoDB Database.
* Java Module to process logs from MongoDB Database to MySQL Database.
* Processed data will be visualized by the analysis module.
* Visualization tool will be used to provide meaningful data representation to the users.

## 3.2. Non Functional Requirements

* The non-functional requirements are the following:
* Scalability : The System should be highly scalable so that it can meet our future requirements.
* Consistency : The data in the logs obtained from logstash and modified , the data in MongoDB, MySql and the visualization charts should be consistent.
* Performance: The graphs should be loaded without any delay as per the requirements.
* WorkLoad: System should be capable enough to handle heavy workloads.

# 4. Design

## 4.1.Algorithm

**DRS-( PART-1)**

* Analyising the CPU usage of all the VM and vHOST.
* Check the vHOST with minimum CPU Usage.
* After finding the vHOST with minimum CPU usage add the new VM to that vHOST.

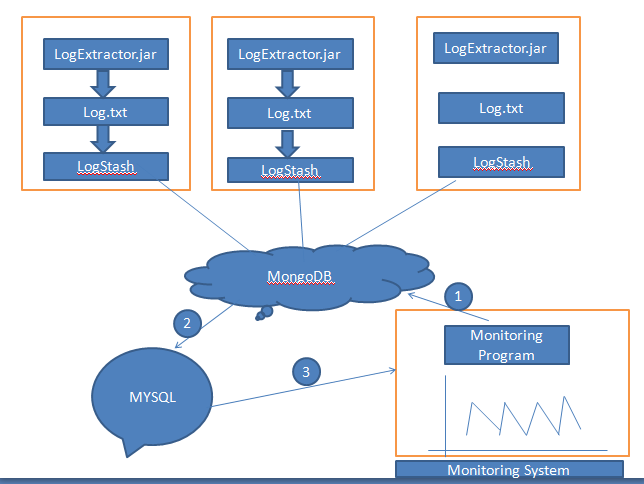
**DRS-(PART-2)**

* Precondition : DRS UpperThreshold set to 70 % CPU Usage
* Unbalance all the vHost by adding the VM to the vHosts till the time vHost reaches the DRS UpperThreshold
* DRS program will kicks in as soon as the the Upper Threshold is reached to any of the vHOST.
* New vHost added into Data Center when the Unbalaced vHost found.
* One by one VM will be migrated from unbalanced vHost to the New vHost till the the time Cpu Usage for the Unbalanced vHost goes below the 70 % Threshold level

**DPM**

* Precondition: DPM Lower threshold value of CPU usage set to 30%.
* DPM program will keep on scanning the CPU usage of all the vHosts.
* Whenever vHost with cpu usage less than lower threshold found, DPM program will kicks in and migrate all the VM from this vHost to the another vHost in the Data Center.
* Finally DPM will remove this under utilized vHost from the Data Center.

## 4.2. Architecture



## 4.3. Key WorkFlows/Components

**Logstash :-**

The logstash are basically used to gather the logs from the Virtual Machine. Logstash basically collects the logs of all the resource usage happening on a particular Virtual Machine.

**MongoDB(MongoLab):-**

MongoDB is basically used for storing the data. It is used to store the logs that are collected by logstash. Mongodb is just used for storing purpose as the data stores is unstructured data.

**MySQL(Setup on RDS)** **:-**

MySQL is used to transferring the data which is stored in MongoDB. The unstructured data that is stored in MongoDB is then converted into a proper structured format in MySQL as required.

**Visualization :-**

The data which is parsed in MySQL is then used to further to get it visualized. For Visualization purpose tools like Google Charts, HighCharts, Tableau, qlikview etc can be used. We have used Google Charts for the construction of graphs using MySQL. We have used php to get the data in real time over the server.

# 5. Implementation

## 5.1. Environment

* Cirius 8 at host 130.65.132.160:
* Host 1 : 130.65.132.161
* Host 2 : 130.65.132.162
* Host 3 : 130.65.132.163
* These hosts are clustered together on a cluster and they are managed by vCluster\_Team04.

## 5.2. Tools and Technologies

Servers

* WAMP

Databases(Made use of Cloud Services for the Storage)

* MongoDB - NoSQL Database We have used MongoLab
* MySQL - SQL Database Setup on AWS RDS

Front- End

* Google Charts
* jQuery
* PHP

Back-End

* Java
* PHP

VMware specific technologies

* VMware vSphere Client - To access the Virtual Machine and vHosts
* VMware VI API

Managing the code and editing

* Eclipse
* GitHub

Ubuntu 32 bit Operating system installed on all VM’s

### 5.2.1. Application  Performance Management

* This application is basically used to monitor performance and availability of all the Virtual Machine and Hosts.
* This is done by using the stats that are gathered from every Virtual machine and Host and measured performance is then visible to the user.

**Framework**

o   User view.

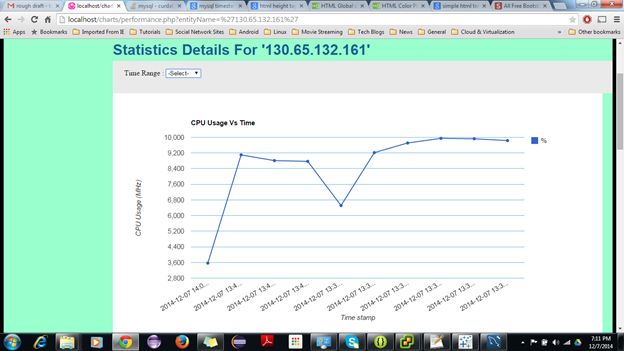
o   Architecture of the Runtime Application.

o   Monitoring the Components.

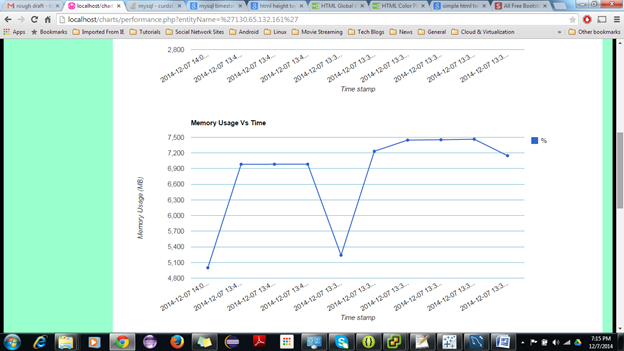
o   Performing the Analysis.

## 5.3. Screenshots

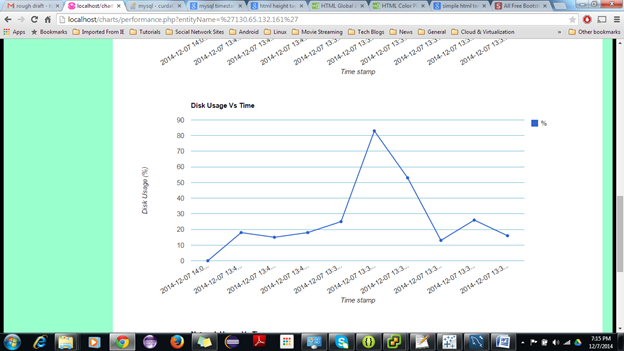
**CPU Usage vs Time**



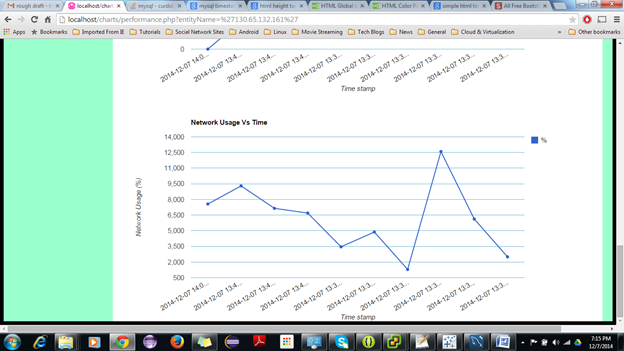
**MemoryUsage vs Time**



**DiskUsage vs Time**



**Network Usage vs Time**



# 6. Assumptions

* The setup is assumed to handle less than 15,000 database connections from the agents.

# 7. Limitations

* The team has setup 2 Virtual Machines and 2 Host. The data collected from all the VM’s and vHosts are put into a log collector and is pushed into MongoLab. The maximum storage provided in free subscription plan is 0.5 GB. Therefore archiving of data at regular intervals becomes necessary.
* We have installed Logstash and we run a jar file which collects data from all the VM’s and vHost’s. In order to gain higher range of real-time values of the performance metrics collected , we run Prime95 in all the VM.
* As we run a java file in our local machine to collect data from MongoLab and insert it into RDS MySQL, Network connection is

# 8. Future Work

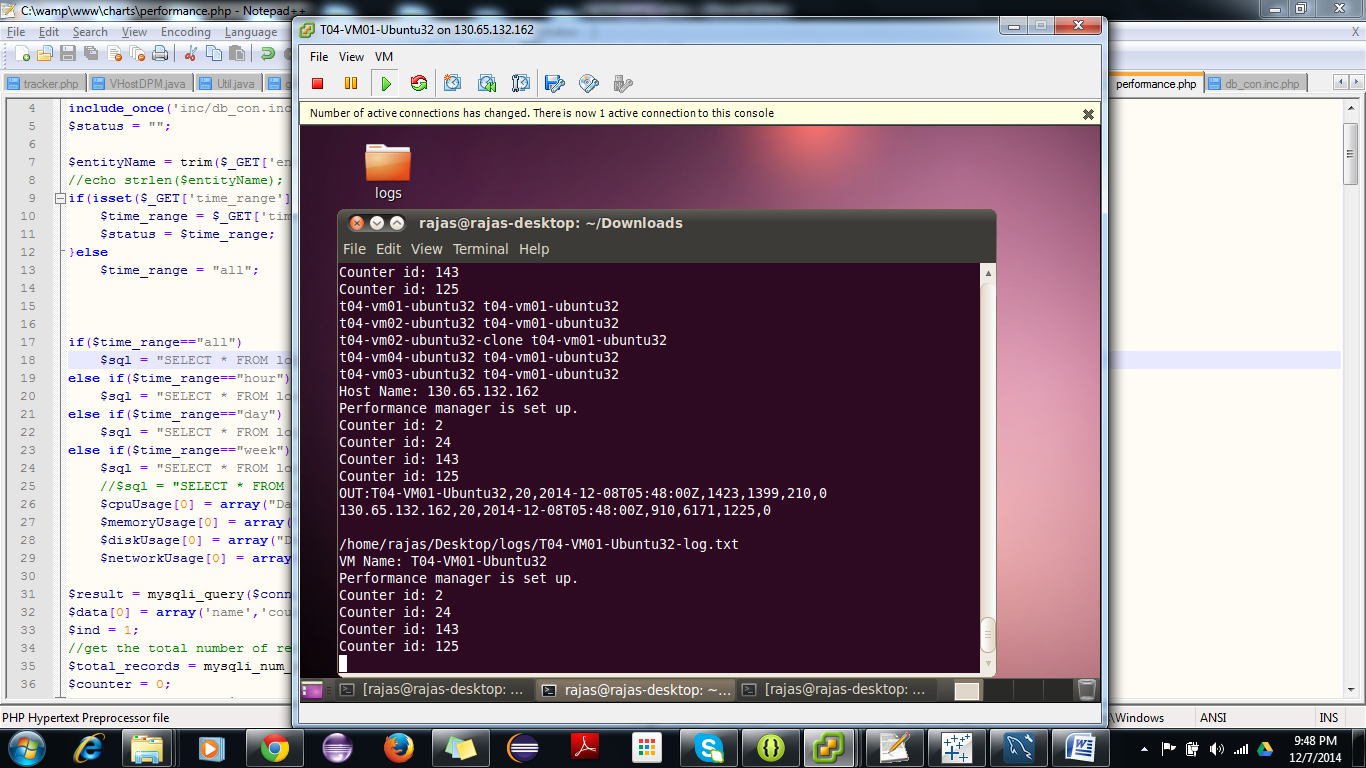
1. We can implement more robust parser by adding more number of performance and metrics.
2. Application can be developed to be more scalable.
3. We can increase the number of hosts and we can use quad core for each individual hosts so that we can increase the number of threads which is responsible for collecting data from the databases.
4. Free subscription  of MongoLab storage limits us from storing more than 0.5 GB data. And, hence a different approach could be taken for the storage.

# 9. Individual Contribution

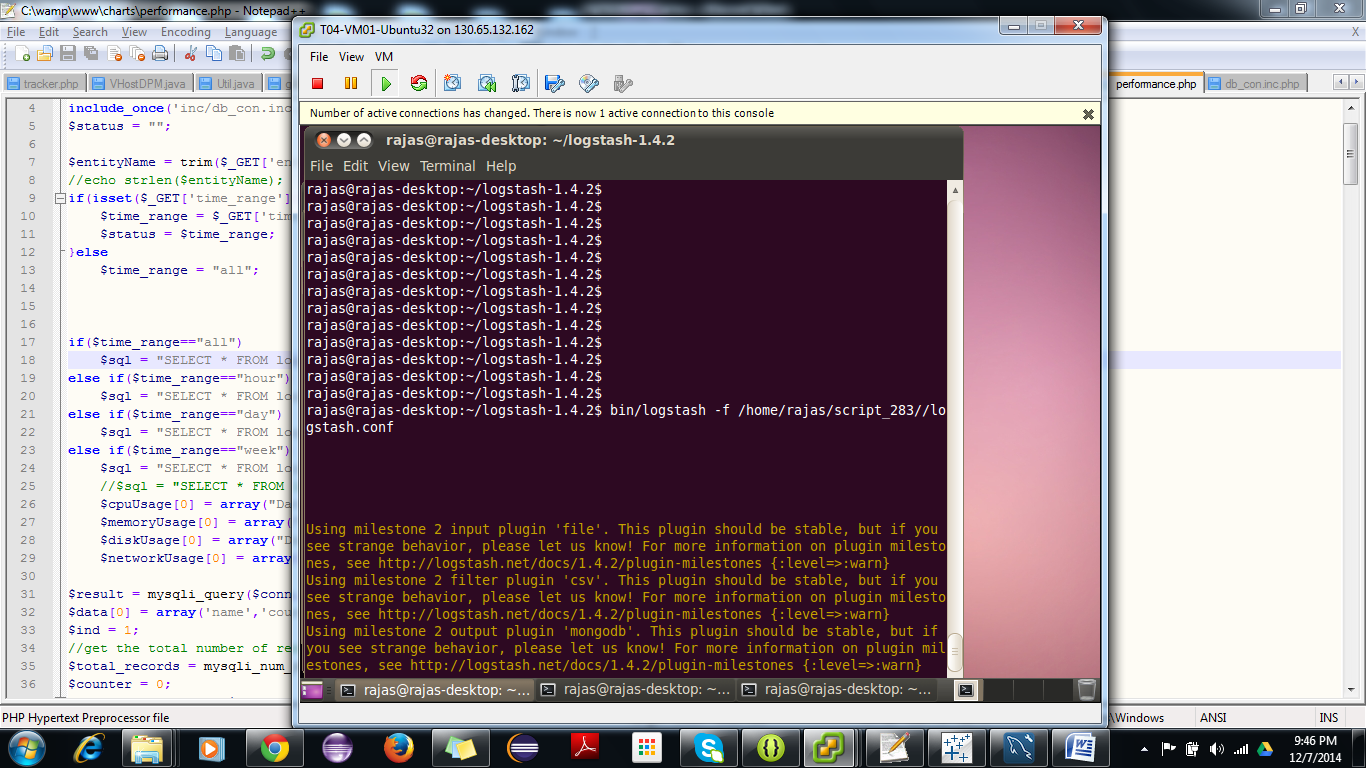
|  |  |
| --- | --- |
| **Team Member Name** | **Contribution** |
| Vaibhav Tupe | Project Planning |
|  | DRS1 and DRS2 |
|  | Performance metrics extraction and Logstash |
|  | UI development using Google Charts  Report |
| Poornima Peddinti | Project Planning  Data Pooling using Logstash  Data Transfer from Logstash to MongoDB  Data Transfer From MongoDB to MySQL  Report |
| Shreya Subhlok | Project Planning |
|  | DB Design |
|  | DPM |
|  | Testing  Report |
| Rajas Hegiste | Project Planning |
|  | DB Design |
|  | DPM  Testing  Report |
|  |  |

# 10. Installation and Execution manual

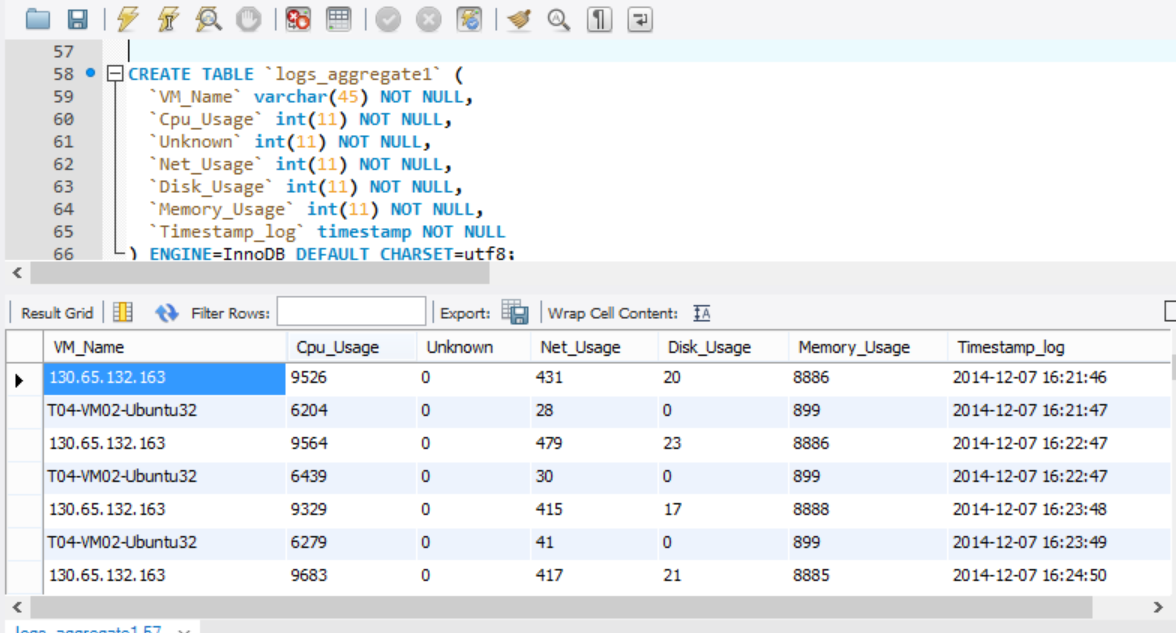
* **Installations**
  + Install the VMware vSphere Client, Eclipse, MySQL, Mongodb and WAMP server in the system.
* **Execution**
* Step 1:Running LogExtractor Jar on each VM which will write logs to the Log.txt file.

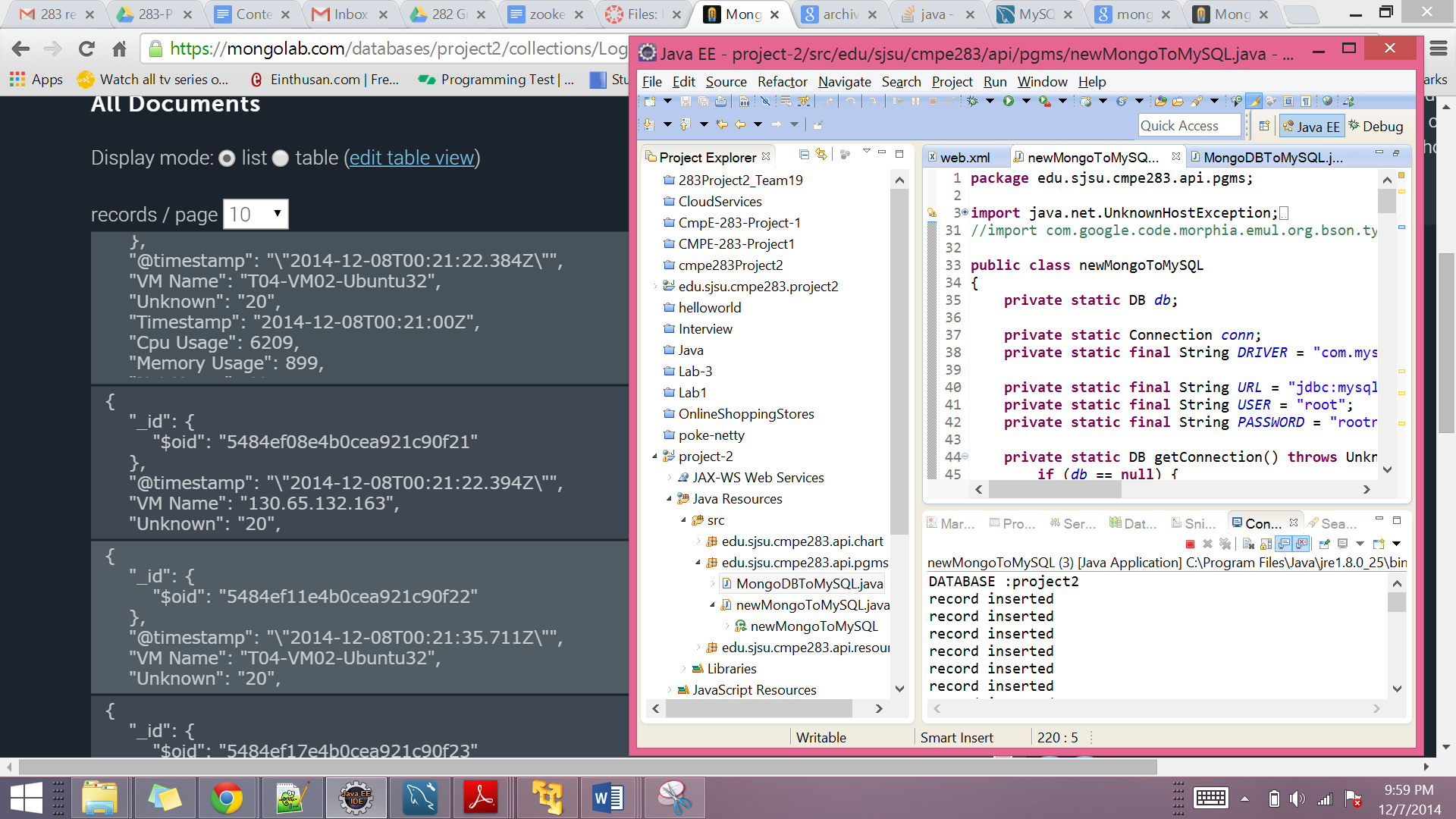


* Step 2: Running the Logstash with logstash.conf file as input to the Logstash

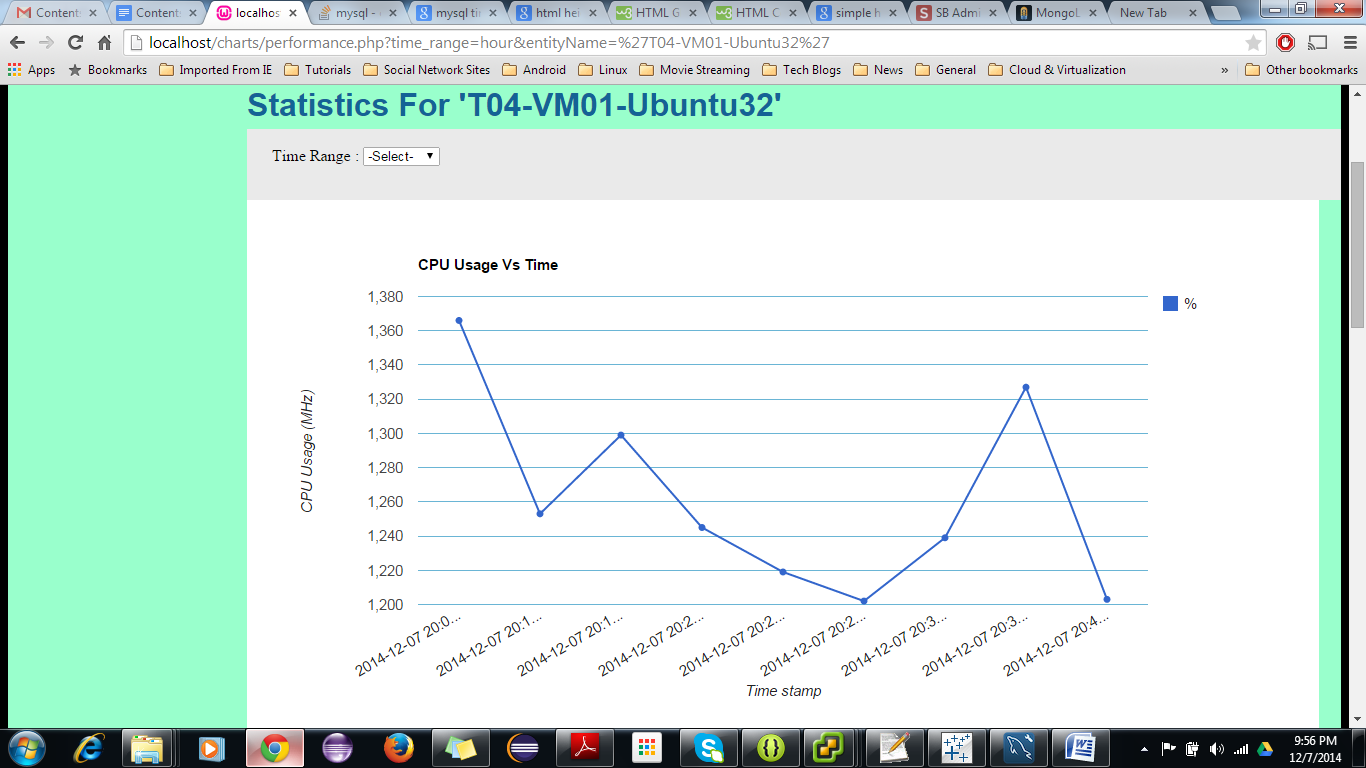


* Step 3: Running Mongo DB to MySQL java program on monitoring system.





* Step 4: Viewing the Visualization Charts in Browser



# 12. Testing

## 12.1. Unit Testing

**Log Collection and Visualization**

The team performed unit testing on each of the components mentioned below.

1. VM log collection

The performance metrics for all the VM’s are being collected properly or not was tested.

1. vHost log collection

The performance metrics for all the vHost’s are being collected properly or not was tested.

1. Logstash Collector

All the performance metrics which were collected properly from using logstash collector or not was tested.

1. MongoDB Insertions

The collection of the performance metrics was converted into proper JSON format and inserted into MongoDB so that the aggregated results could be stored in MySQL or not was tested.

1. MySQL Insertion

The JSON Objects collected from MongoDB was aggregated and inserted into MySQL database. Whether these operations were happening as expected or not was tested.

1. Visualization Charts

Rolls up’s for 1 minute,5 minute, 15 mins, 1 hour and a day’s were being represented on the graph as expected was tested.

**DRS**

Overloaded the VM by running Prime95 and adding the VM to the vHost which resulted into increased CPU Usage.

**DPM**

   Kept one vHost with only one idle VM in it,Which resulted into under Utilization of vHost.(CPU Usage < 30 %)

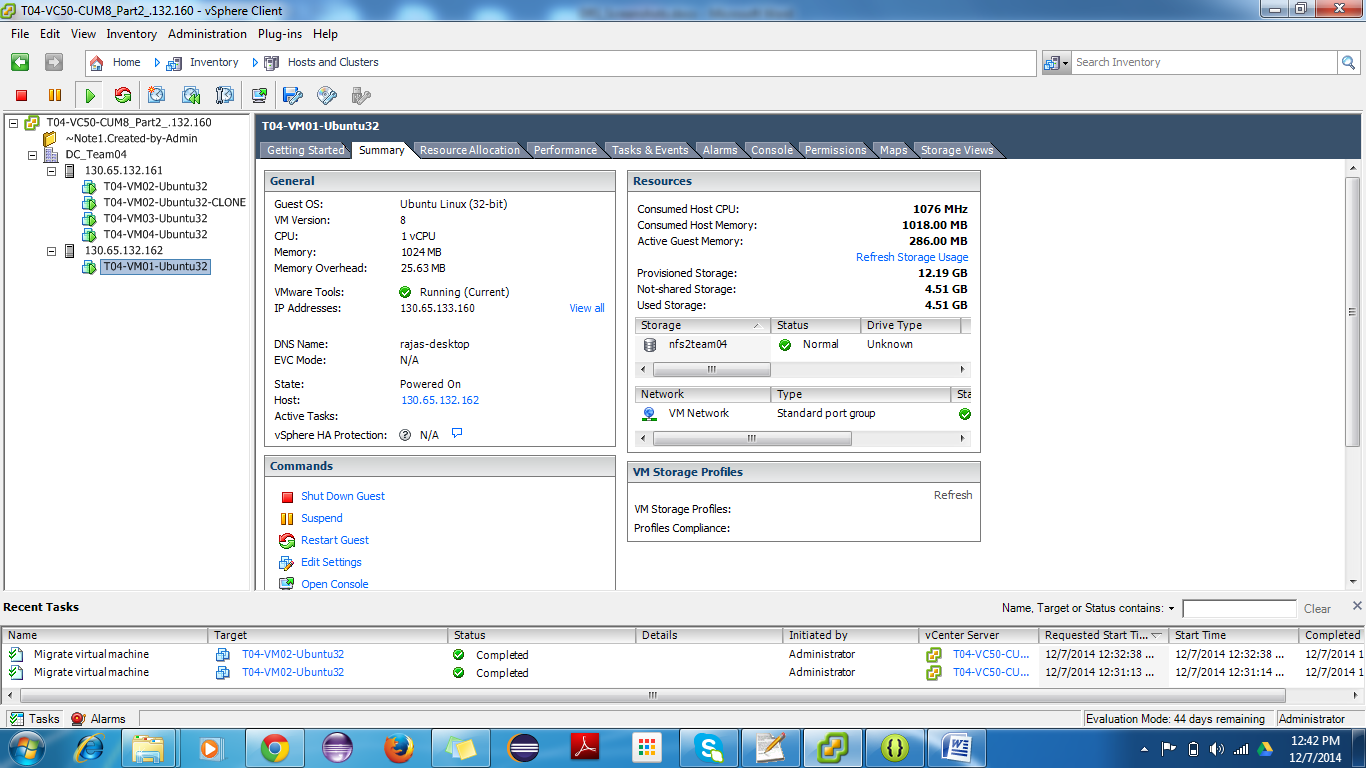
## 12.2. Integration Testing

Integration testing was done by integrating all the modules mentioned above in the Unit Testing section and testing it from end to end. All the functionalities to be implemented as per the given requirements were checked and tested.

# 13. Screenshots with annotations

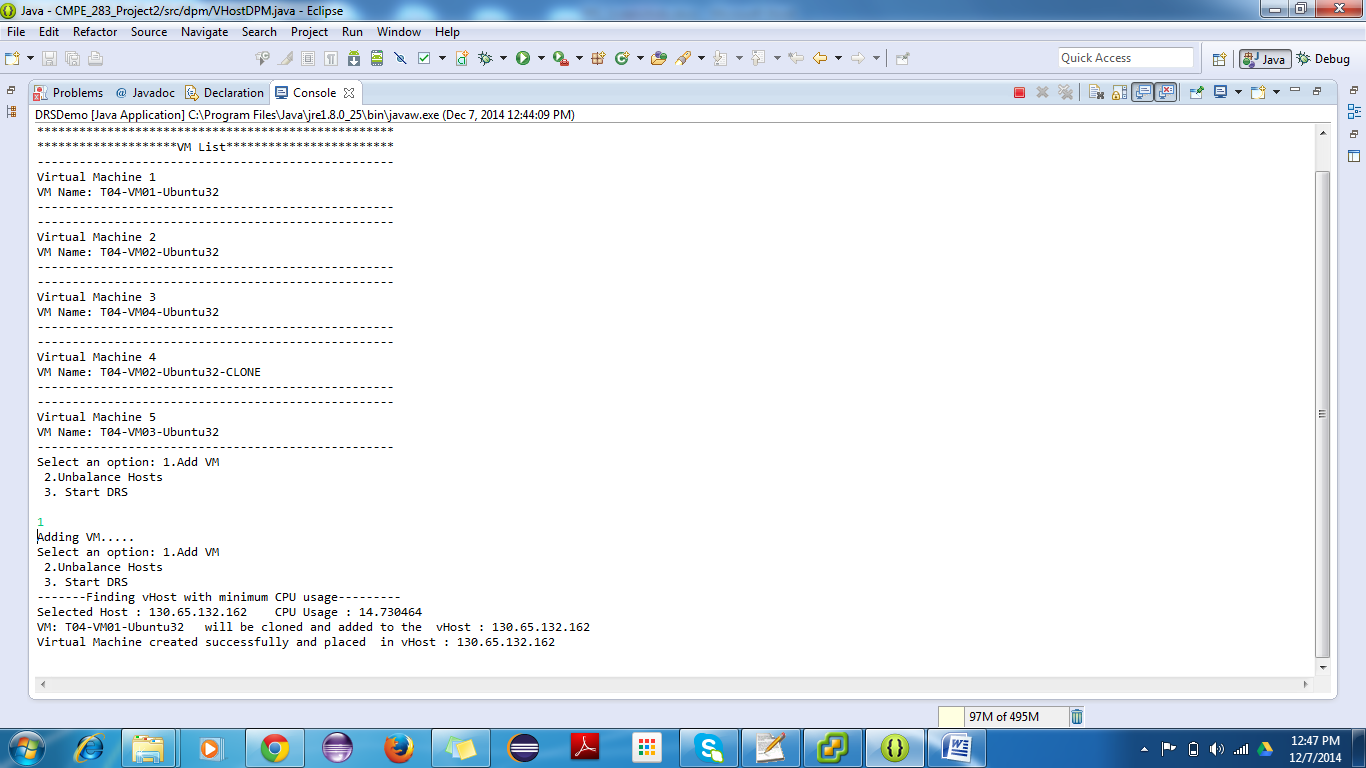
**DRS**

Initial inventory and Data Center status

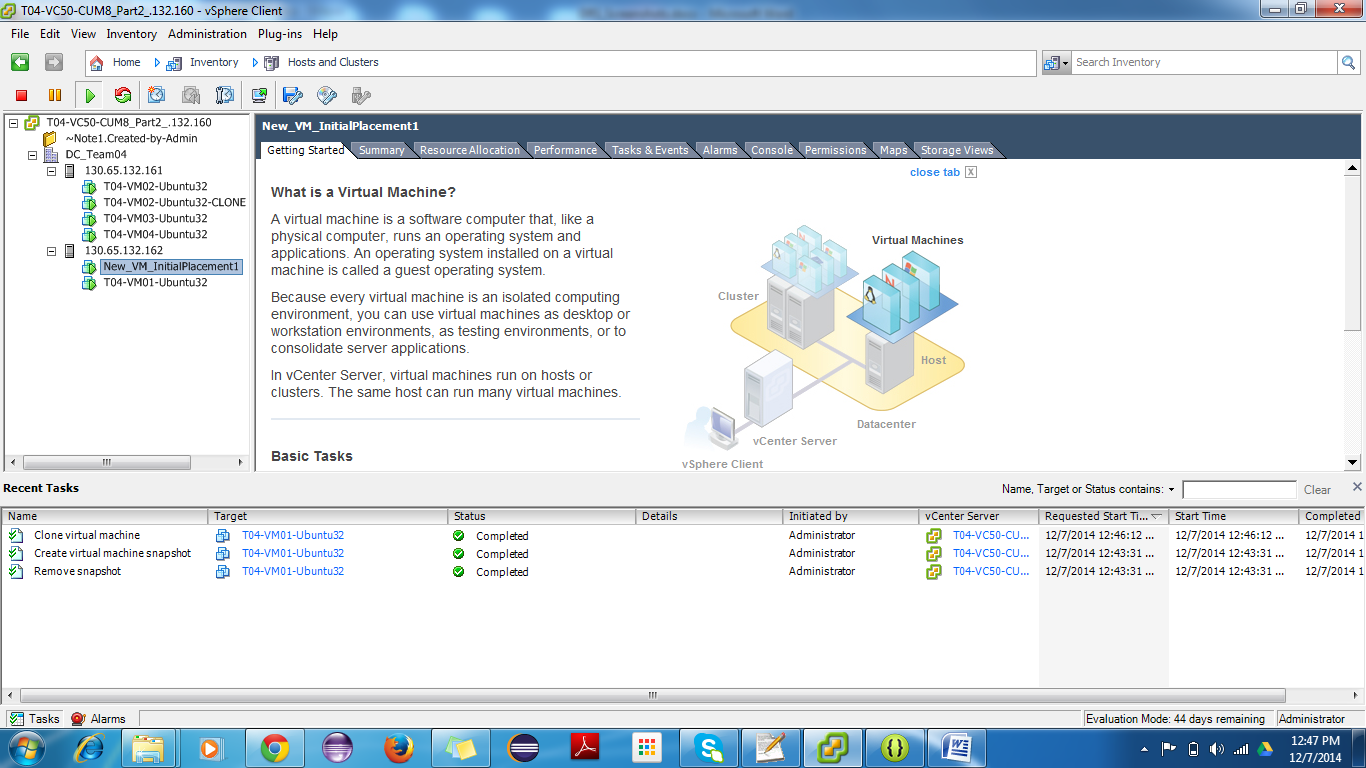


**--DRS PART 1:**

Initial Placement of VM based on Cpu usage of vHost

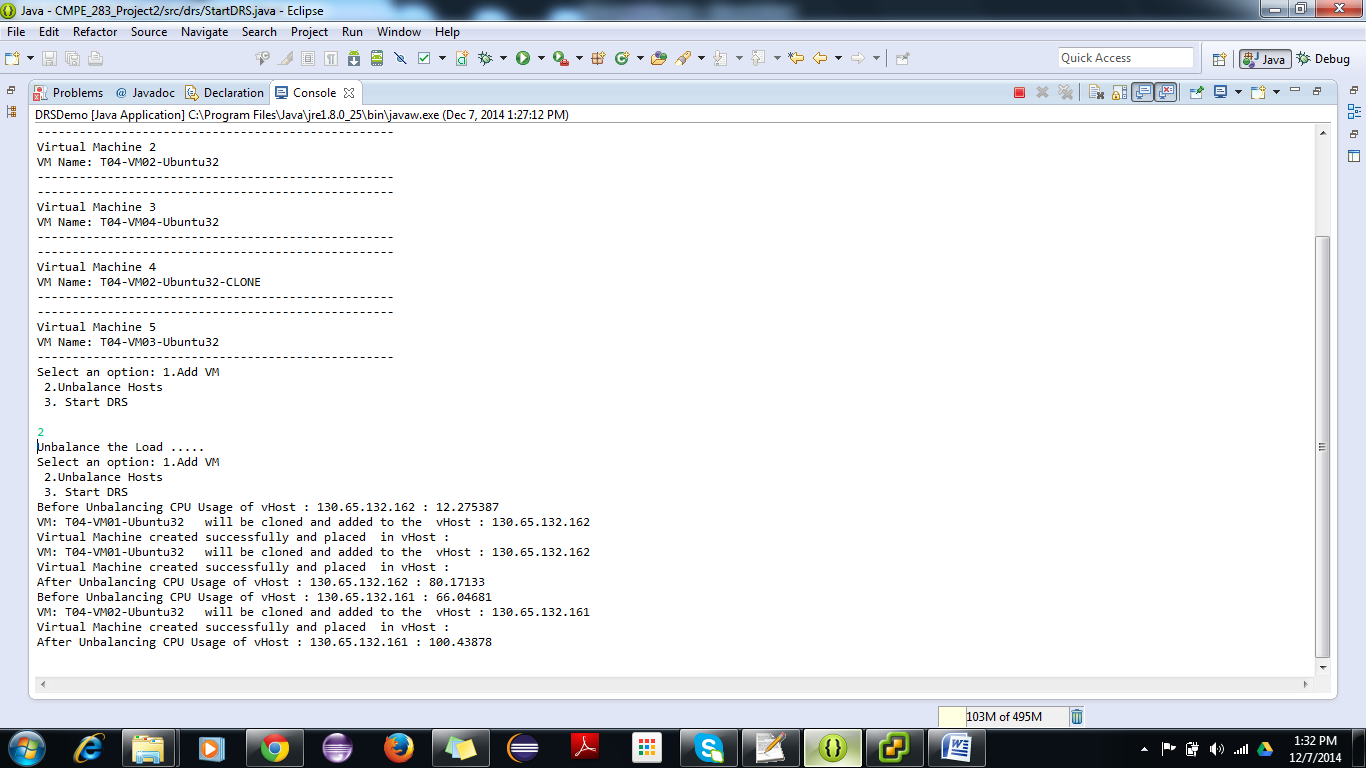


-Placement being done is shown in the image below.

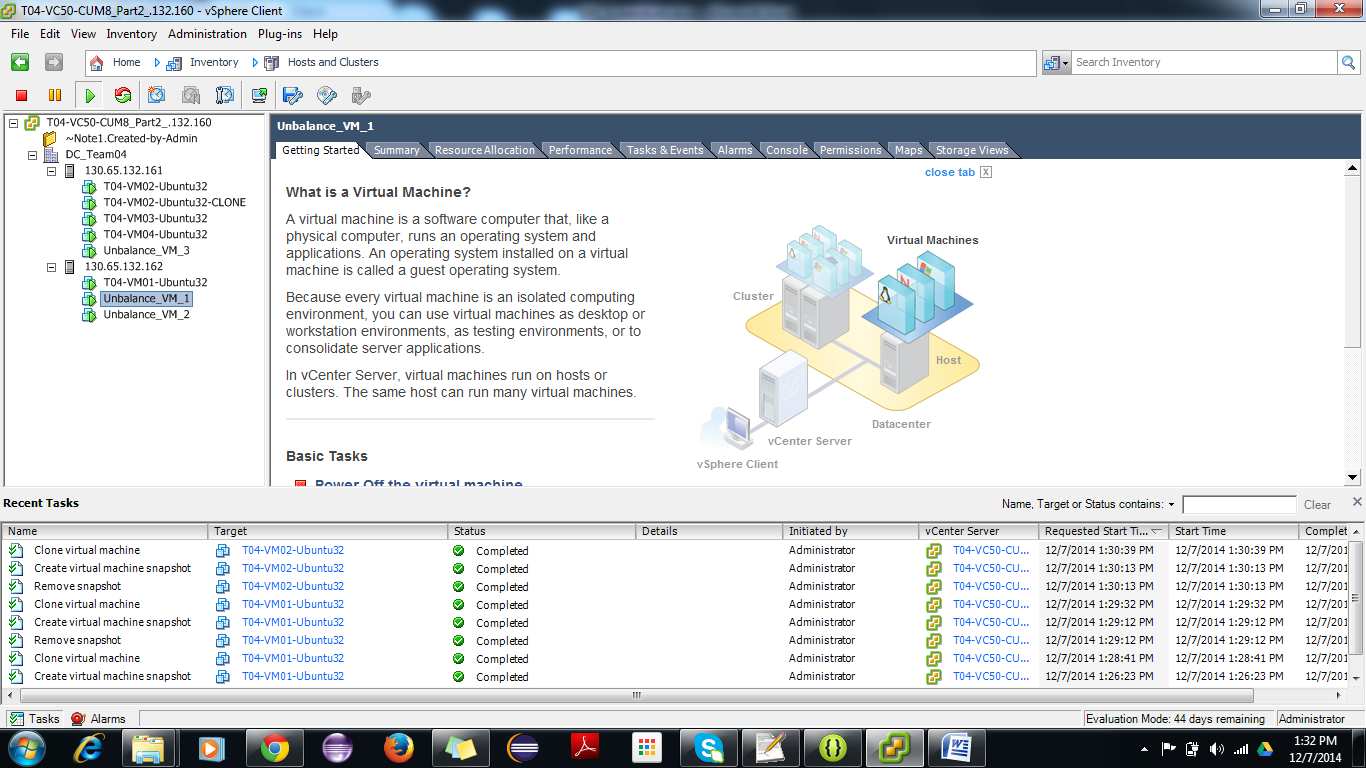


**DRS PART 2:**

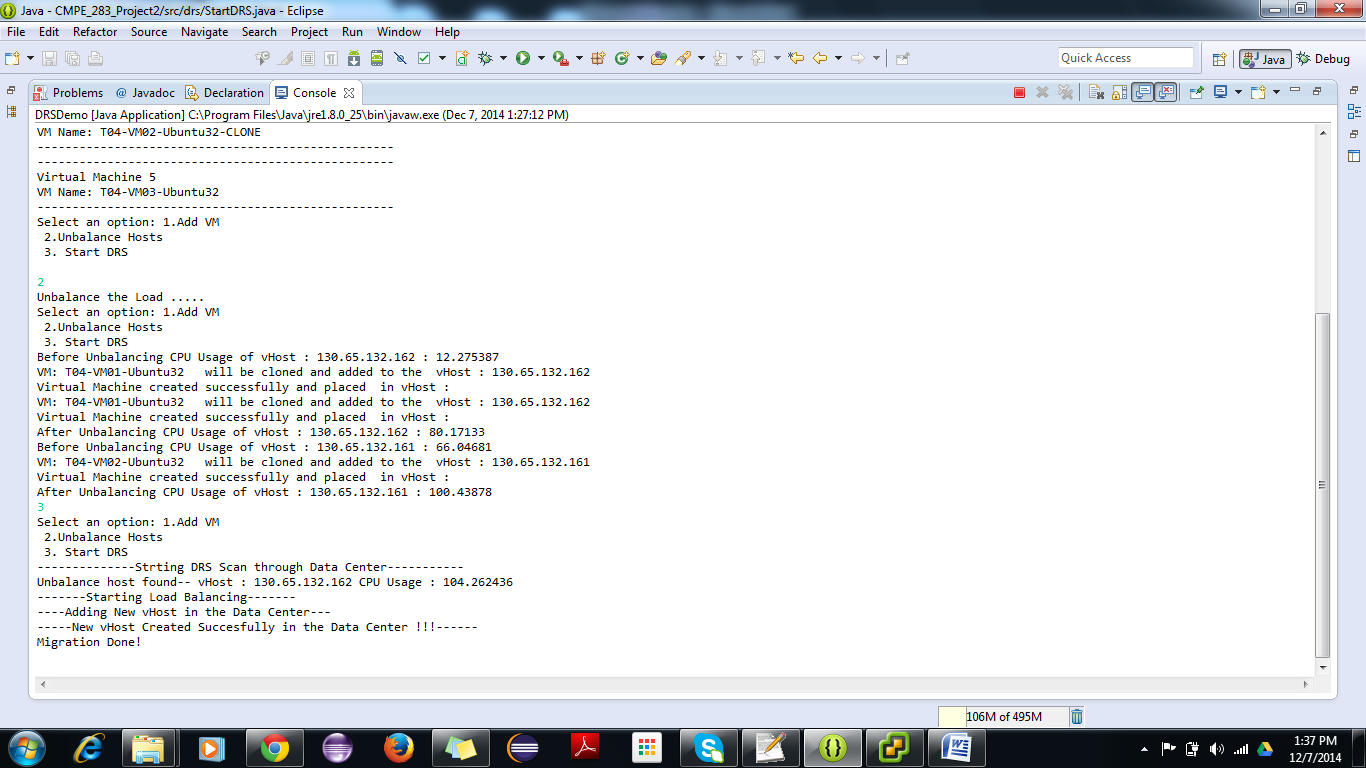
Make Host unbalanced by adding VM.



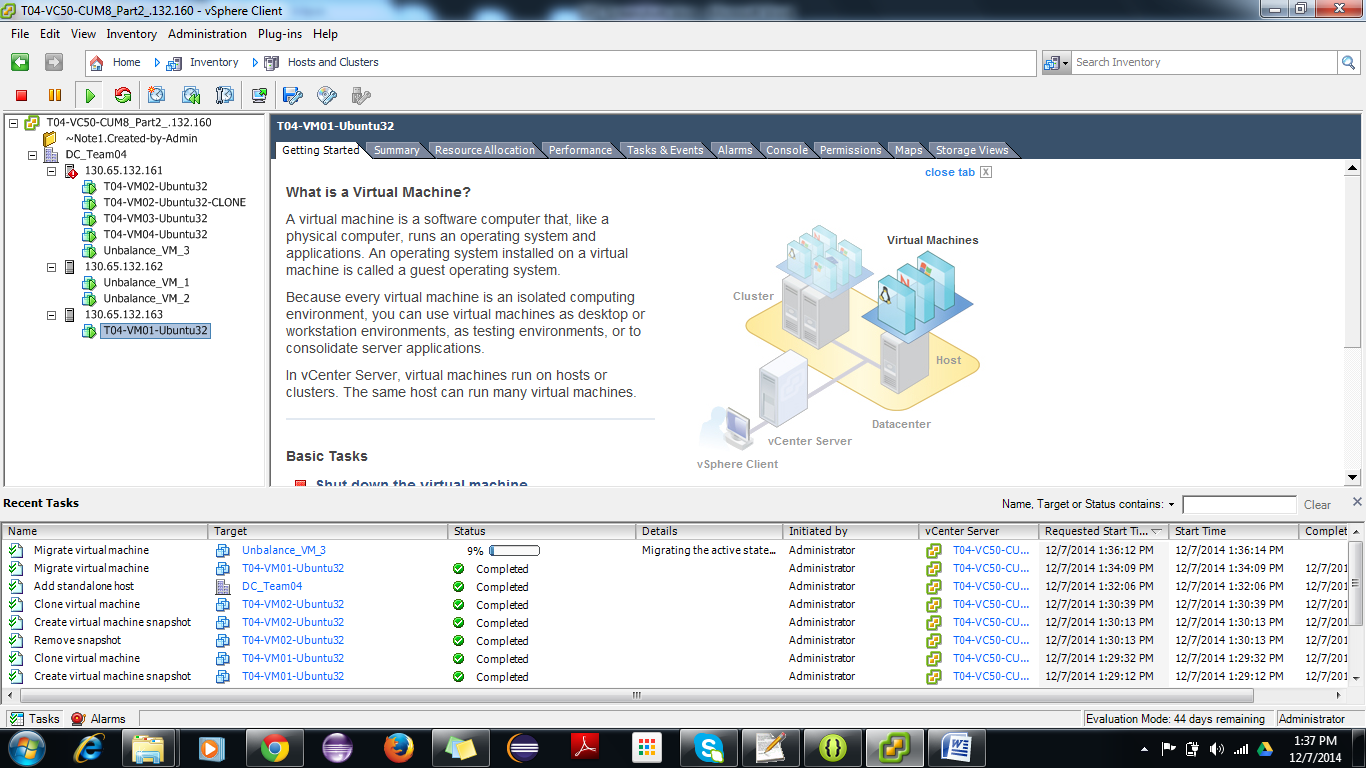
--VM being added to the vHost’s to make those unstable



--Running DRS now to add new Vhost and balance the load

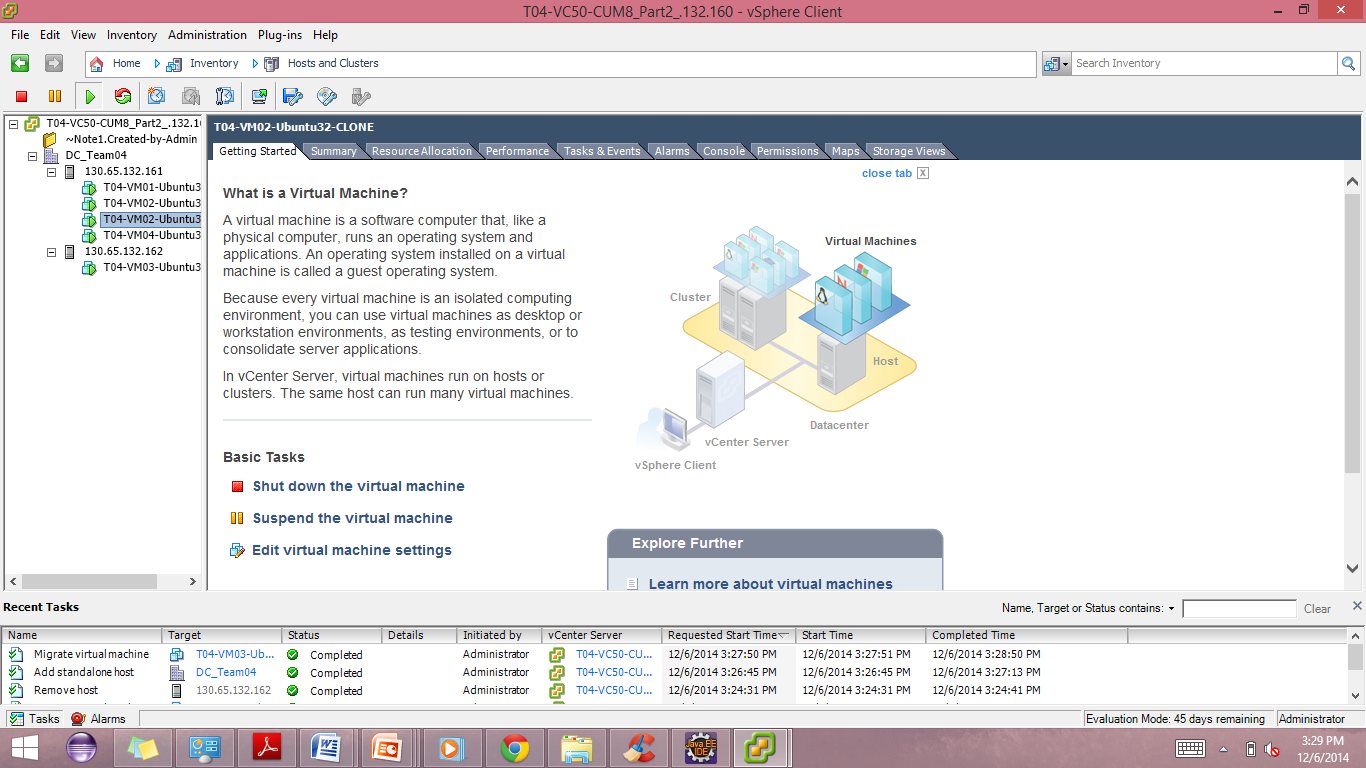


--DRS adds the new vHost and migrates the VM from the unbalanced vHost

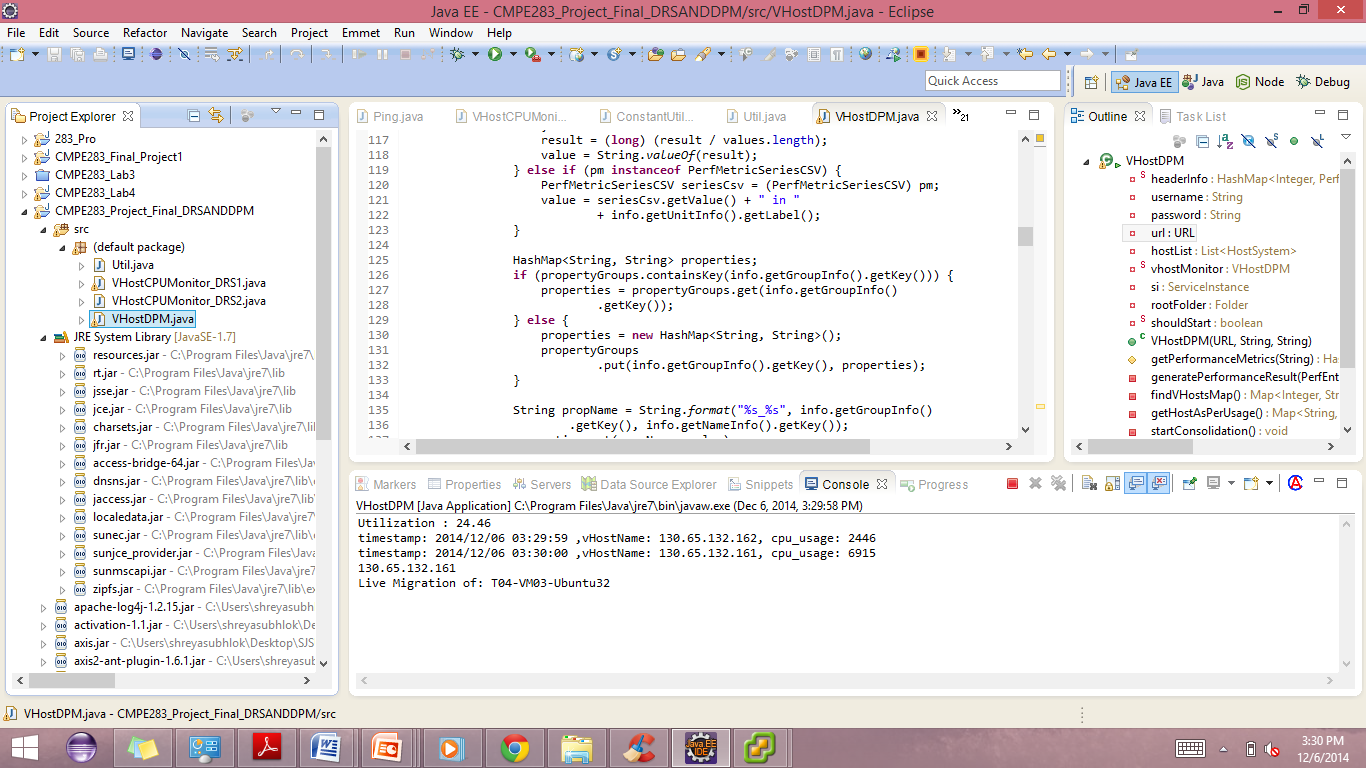


**DPM**

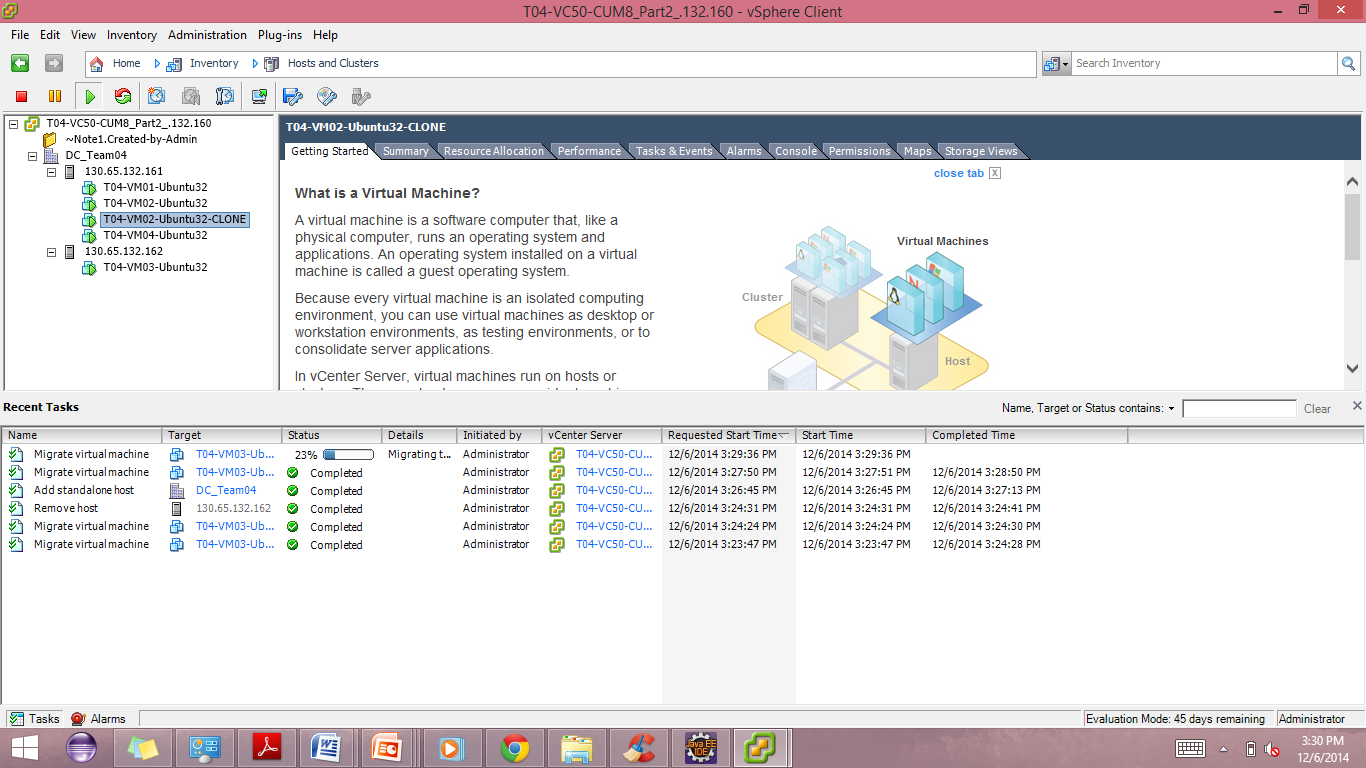
-- Initial state with 2 vHosts one with usage below 30%

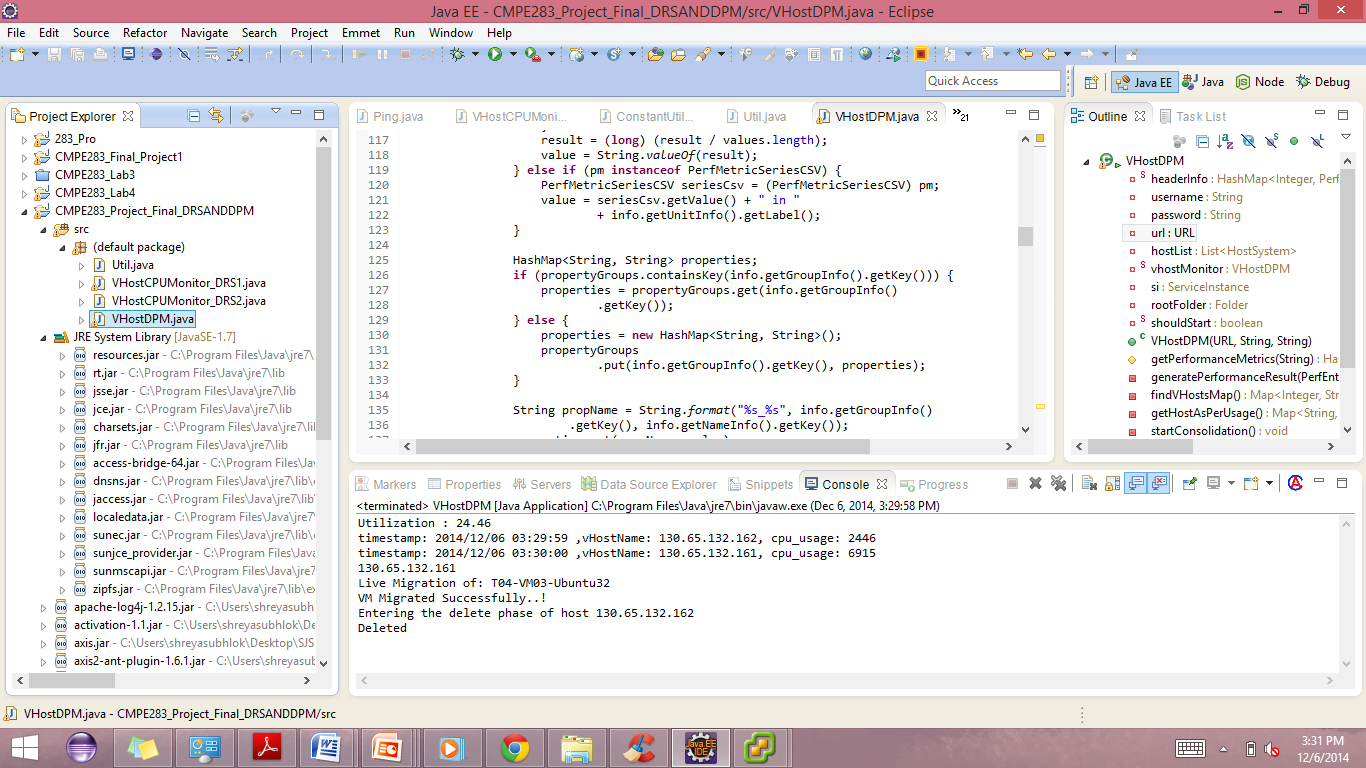


DPM Program kicks in and finds out vHost 2 is with cpu usage less than 30%

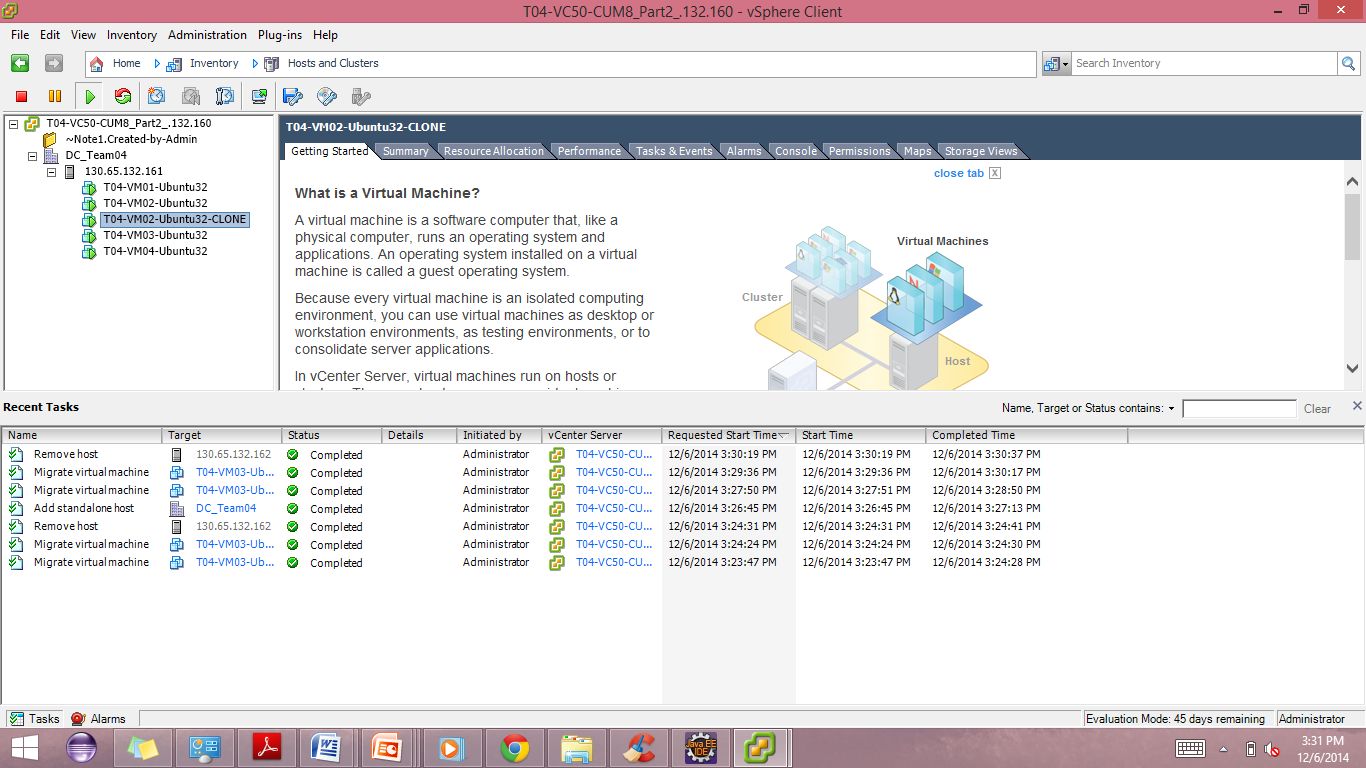


DPM starts migrating VM from vHost2 to vHost1





DPM migrated VM from the under-utilized vHost and removes that vHost from the inventory



# 14. Conclusion

Successful implementation of a complete framework to send the logs of VM’s and vHost’s , displaying each metrics against the time series(1 minute, 5 minute, 15 minutes, 60 minutes and a week’s data) using Google charts were done.

Also, successful implementation of DRS1,DRS2 and DPM was done with respect to the given requirements.

## 14.1. Lessons Learned

1. Handling Large volumes of Data
2. Familiarity with the VMware API’s by implementing DRS1,DRS2 and DPM
3. Load balancing of various resources
4. Hands-on LogStash and various visualization tools
5. Logs analysis gave us an insight on the behaviour of the VM’s.

## 14.2. Challenges

1. Developing the framework which gives a better performance, scalability,resource utilization was quite challenging.
2. Deciding on  memory and performance tradeoffs for the application during the design phase was quite time consuming.
3. Initially, working with Logstash was difficult and understanding the way it functions was time consuming.
4. Integration of the code developed consumed some time.

# 15. References

[1]   <http://www.mongodb.org/>

[2]   <http://en.wikipedia.org/wiki/Application_performance_management>

[7]   http://www.highcharts.com/

[3]   http://www.vmware.com/pdf/esx\_lun\_security.pdf

[4]   http://vmware.com/support/developer/vc-sdk/visdk400pubs/ReferenceGuide/

[5]   http://www.vmware.com/pdf/esx\_lun\_security.pdf

[6]   http://docs.mongodb.org/ecosystem/tutorial/getting-started-with-java-driver/