

**CMPE 283 –VIRTUALISATION TECHNOLOGIES**

**PROJECT- 1**

Advisor: **Dr. Simon Shim**

**Submitted by**

Anita Gnanamalar Jebaraj

SJSU ID: 009305322

**INDEX OF CONTENTS:**

1. Introduction……………………………………………………………………3

2. Background………………………..…………………………………………..3

3. Requirements…………………………….…………………………………….4

4. Design………………………………………………………………………….5

5. Implementation……………………………...…………………………............7

6. Answers to Questions……………..………………………………………….16

7. Discussion…………………………………………………………………….17

8. Conclusion……………………………………………………………………17

9. References……………………………………………………………….……18

**1. INTRODUCTION**

**Goal:**

The goal of the project is to develop a prototype of availability manager, which monitors the liveliness of the virtual machines at a host and if that host fails, a mechanism should be devised to activate that host again and restart the virtual machine. It helps in gaining experience in using multiple hypervisors and the corresponding management servers, explore the capabilities offered by the components and their APIs. It also helps in learning about the various issues that would arise out of interoperability.

**Objective:**

The objective for the development of the availability manager is to gain hands-on experience with the hypervisors and their management servers. It provides the opportunity to explore the capabilities offered by the components and their APIs and to learn about the various issues that would arise out of interoperability.

**Need for the Availability Manager:**

Availability Managers can be used to monitor the state of the virtual machines, recover the virtual machines and hence considerably decrease the downtime of the systems. Manual process of monitoring the virtual machines requires more manpower and time and therefore the process cannot be efficient. Availability management process is to define, plan, evaluate and improve all aspects of the availability services.

**2. BACKGROUND**

Disaster recovery systems are vital to every organization in order to track the progress of the various processes that will be held everyday. Manual monitoring of the functioning of the systems may require more resources, manual effort, cost and is difficult to maintain, hence the organizations switched to developing disaster recovery systems in various forms. Traditional disaster recovery processes are expensive, as it requires identical hardware configuration at the recovery site, which in turn involves purchasing new machines to maintain the state of the virtual machines, this is not feasible in most the cases and hence an alternative easily implementable solution is required. Disaster recovery plans should include preventive measures, detective measures and corrective measures. The availability manager is a disaster recovery system, which can be used to identify the status of the virtual machines by pinging the virtual machines periodically, when the virtual machine stops replying to the ping, it triggers the message that a problem has occurred. In order to correct the problem, the snapshots of the virtual machines are taken when it is active and the same is used to revert the machine to its previous working state. The same procedure can be used to check and maintain the status of the host machine. This system is easy to develop, involves fewer resources and is cost effective.

**3. REQUIREMENTS**

**Functional Requirements:**

The functional requirement of the project is to develop an availability manager by using VMware Infrastructure API, which monitors the status of the virtual machines and recovers the machines in times of failure by using the snapshots. VI API provides the complete set of libraries to maintain the Vmware Infrastructure and vSphere. The APIs are built on top of web services interfaces built on top of WSDL in VI SDK. Therefore in order to use the APIs the VI SDK should be installed first along with JAVA. The APIs provides access to the vSphere management components- the managed objects that can be used to manage, monitor and control the life cycle operations of the virtual machines and other virtual infrastructure components like data stores, data centers, networks etc. The vSphere web services SDK was developed to help the developers to manager the Vmware vSphere components available on Vmware ESXi and Vmware vCenter server systems.

**Non-functional Requirements:**

1. Availability: The availability manager system ensures high availability of the virtual machines. Snapshots are taken periodically and updated in the cache; hence whenever there is a failure in the virtual machines, they are reverted back to the previous state by using the snapshot.
2. Load Balancing: The availability manager is designed to recover the failed virtual machines. The system must be able to proceed with the recovery process irrespective of the load of the system.
3. System Recoverability: The recoverability of the system is designed by taking snapshots for every 10 minutes. Having the current updated snapshot in the cache ensures that the machine can be reverted to its previous state in times of failure.
4. Simulation of Disaster: In order to develop and test the working of availability manager, the host and the virtual machines should be simulated to fail to test the scenario. Disabling the Network Interface card can do this and hence the ping to the machines will not work. This is one scenario, which can be used to simulate the failure of the machines.
5. Fault Tolerant: The availability manager that is developed should be fault tolerant, when the state of the system is reverted back to its available status by using the snapshot the previous working state should be obtained. In order to achieve this, before taking the snapshot of the virtual machines, the state of the vm should be checked to ensure it is in the proper working state.

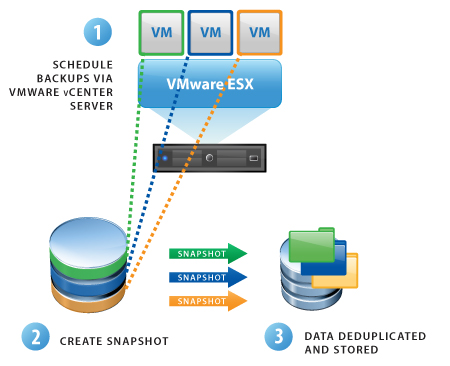
**4. DESIGN**

****

**Figure 4.1: Virtual Datacenter Architecture**

VMware Infrastructure is a set of virtual components, which can be used to virtualize the entire IT infrastructure. Using the VMware Infrastructure, the IT components can be shared, without the need to worry about the underlying hardware. In order to build a virtual data center, the Datastores, networks, hosts, clusters and resource pools are required. These components are essential for the resource sharing ability in the virtual environment.

* Datastores - Datastores are the storage resources, which are the virtual representations of combinations of underlying physical storage resources in the data center.
* Networks - Networks in the virtual environment connect virtual machines to each other or a machine in the physical network outside of the virtual data center.
* Virtual Machines - Virtual machines are designated to a particular Host, Cluster or Resource Pool and a Datastore when they are created. The different states of the virtual machine are powered on, powered off, suspended and idle state. Virtual machines consume resources similar to electrical machines, they use electricity or power only when it is in the powered on state and consumes no power when it is in the other states.
* Hosts, Clusters and Resource Pools –These are the computing and memory resources in a virtual data center. When multiple machines are grouped together it becomes a cluster. The number of machines in a cluster does not matter and it can be added or removed without affecting its underlying properties.



**Figure 4.2: Process of disaster recovery done by Availability manager**

The availability manager works by taking snapshots every 20 minutes and updates the back up cache with the new snapshots that are taken. Every time the new snapshots are taken it deletes the older backups to avoid wastage of memory space. When a virtual machine fails due to external factors, it stops responding to pings from the availability manager, which in turns triggers an issue in the virtual machines, hence the virtual machines are reverted to their previous state using the snapshots that are available in the back up cache. Whenever the Virtual machines are switched off alarms are generated, which helps in getting alerts regarding the change in status of the virtual machines.

**5. IMPLEMENTATION**

The implementation part of this project makes use of VSphere Vclient environment that contains:

* Hosts, clusters, resource pools virtual machines, datastores and networks.
* Virtual Infrastructure JAVA API by VMWare Infrastructure.

Prior to the development of disaster recovery system, the following steps are prerequisites:

* Creation of two VHosts, which are VMware ESXi servers.
* Creating a Vcenter with Windows 2008 server installed, adding the hosts to the Vcenter and creates virtual machines on one of the hosts.

Implementation of Java API, for automating the process of disaster recovery through availability manager has following steps:

* Pinging to check if the host is alive, if the host is alive take a snapshot of the host, otherwise snapshot is not created.
* Similarly, check if the VM is alive and take a snapshot of the virtual machine in its present active state, otherwise snapshot is not created.
* Next it is checked if the unavailability of the VM was because of network problem or the VM is manually powered off. If the problem was due to network unavailability then no alarm is generated. If that is not the case, then alarm is triggered and the program exits.
* While pinging to the Host, if the host is not replying due to network problems, then the snapshot, which was taken when the Host was active is used to revert the system back to its active state and the Host machine is switched on.
* In certain cases the virtual machine fails even when the Host machine is active, in that scenario, the snapshot of the virtual machine is used to revert the system back to its active state and the virtual machine is powered on.

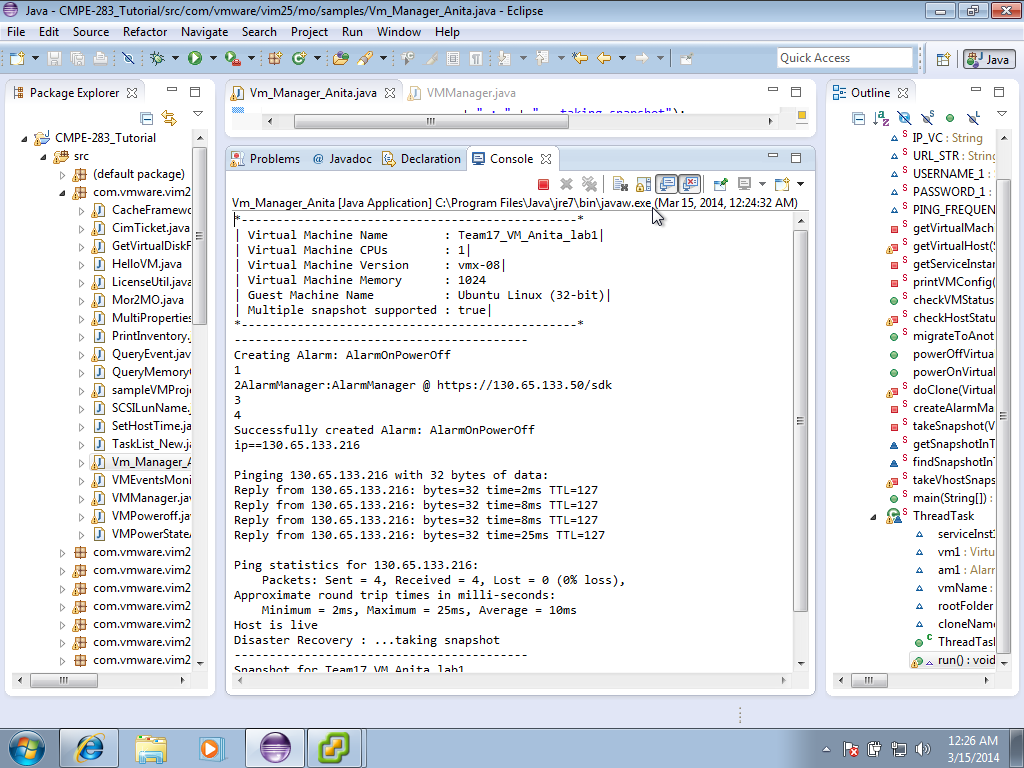
The tools used are Eclipse IDE for integrating the Java API to implement the disaster recovery

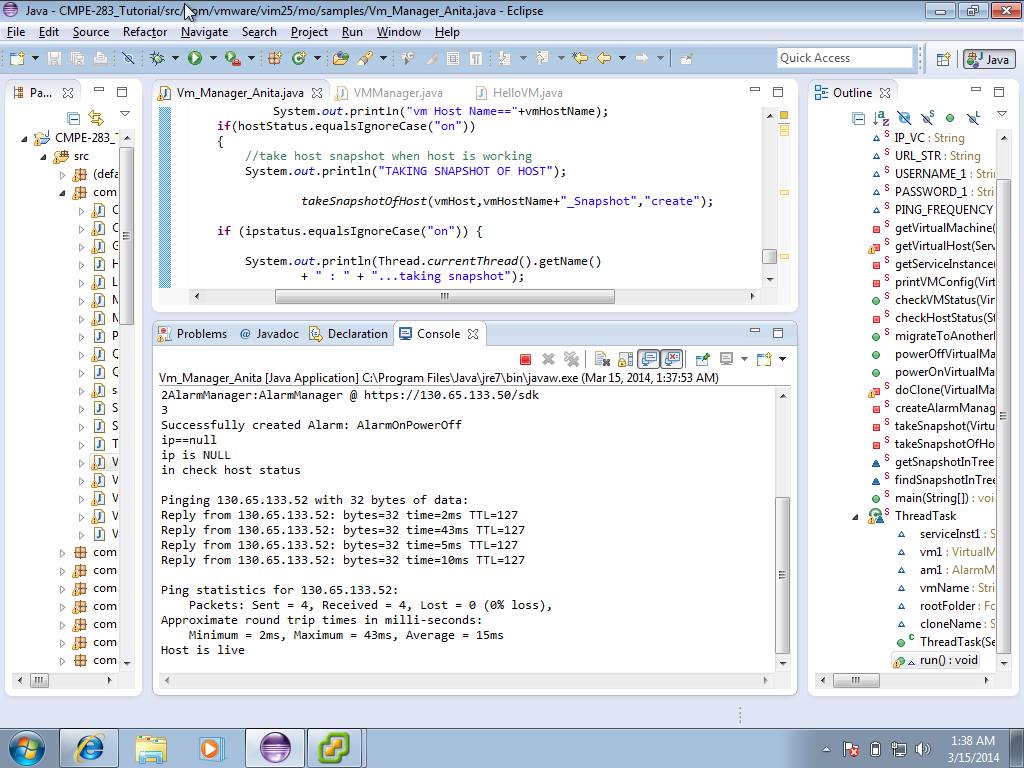
The JAVA class used in the project is VM\_Manager\_Anita, the class contains various functions

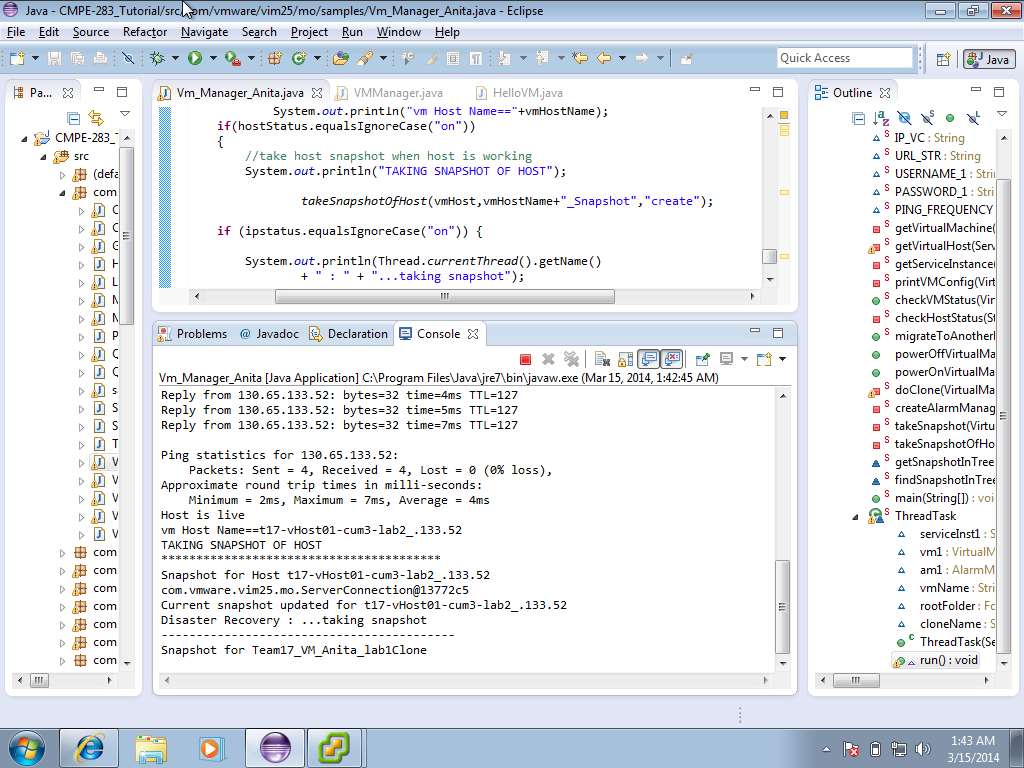
* printConfig() 🡪 prints the configuration of the virtual machine
* checkHostStatus() 🡪 Checks the status of the host by pinging the host IP, returns the value “isReachable” depending upon the status of the host.
* takeSnapshot() 🡪 Takes the snapshot or reverts the snapshot of the virtual machine and the virtual host. The virtual machine name, snapshot name and the operation are given as parameters. Depending upon the operation, which is either create or revert, the corresponding operation is performed.
* createAlarmManager() 🡪 It creates the alarm for the virtual machine. When the state of the virtual machine is changed, the alarm is triggered

**Screenshots:**

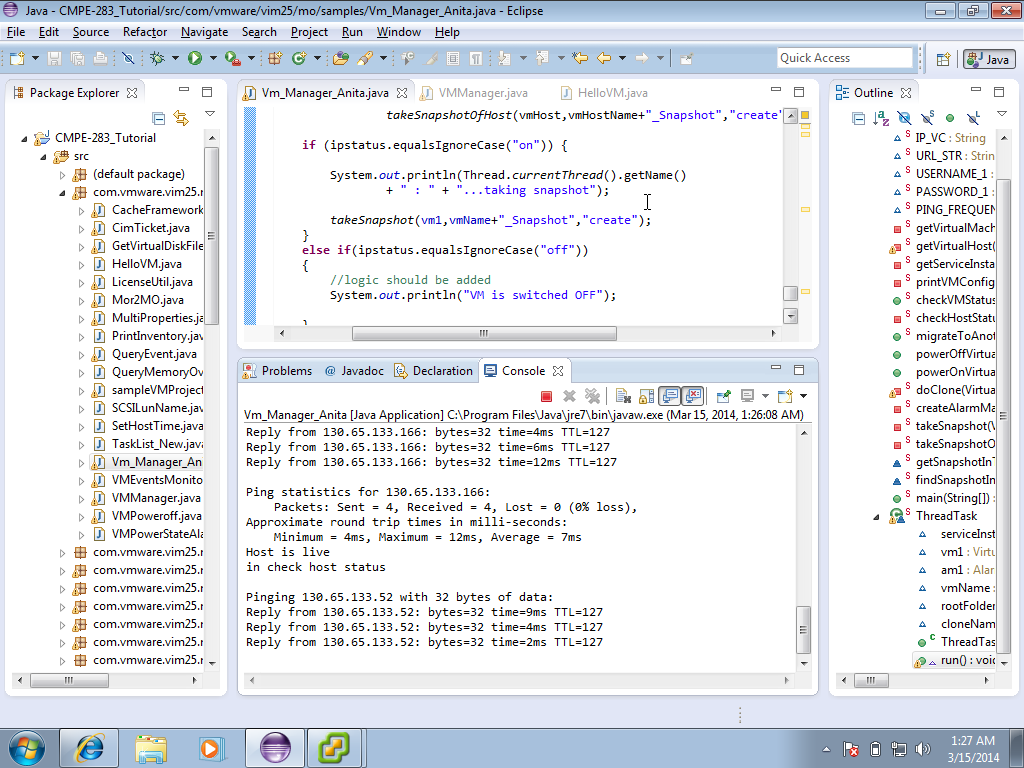
**Step-1:**  In this step the statistics of the Virtual machine is obtained.

****

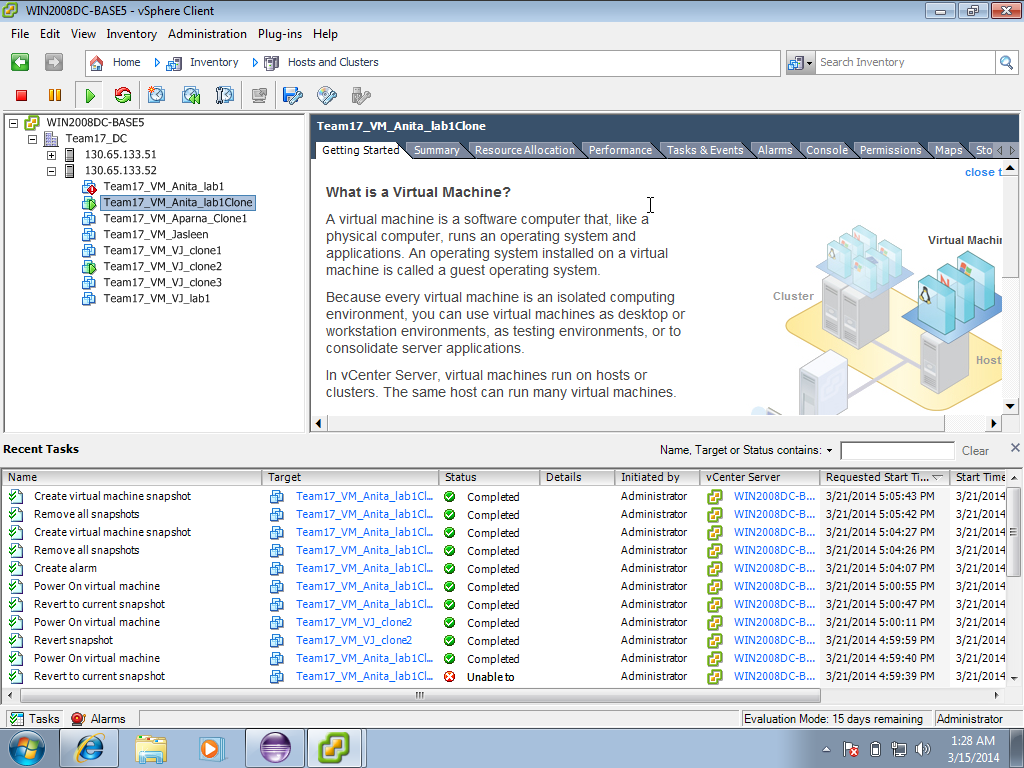
**Step-2:** Pinging the vHost IP addres (130.65.133.52) to check the status of the Host. If the host machine replies to the ping, it means the Host machine is alive.**Screenshot3:** On successful reply from the host, snapshot of the host is taken to update the cache, which has to be done every 10 mins.

****

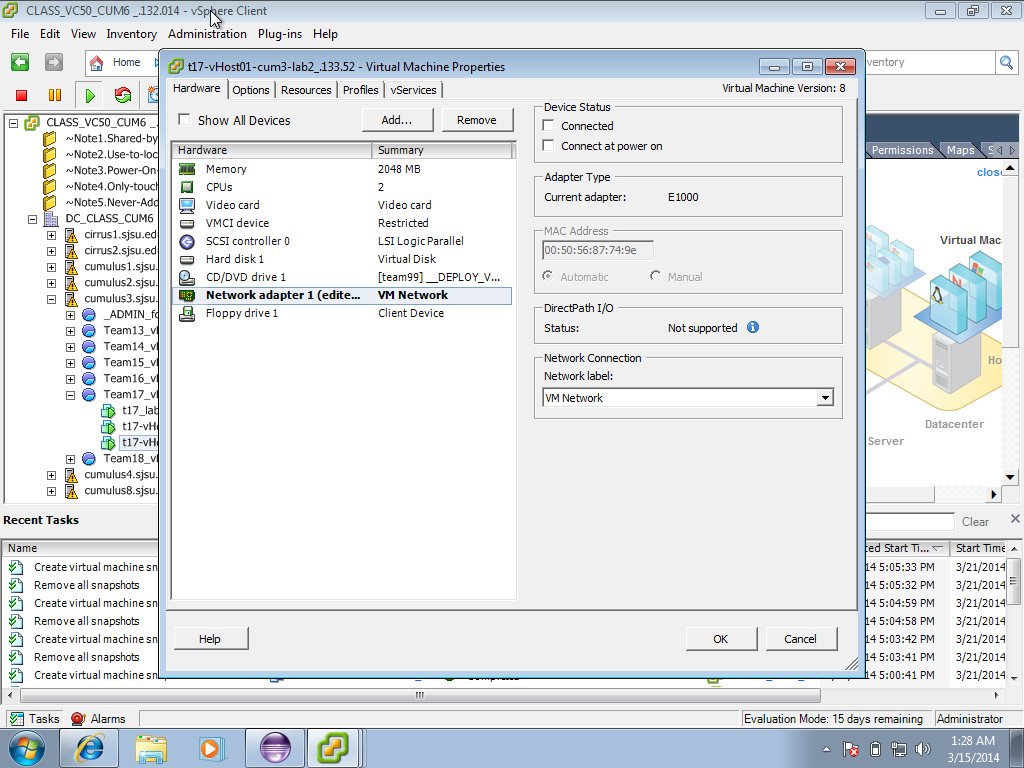
**Screenshot-4:**  After the assurance the host is alive, the virtual machine is pinged to check its status.



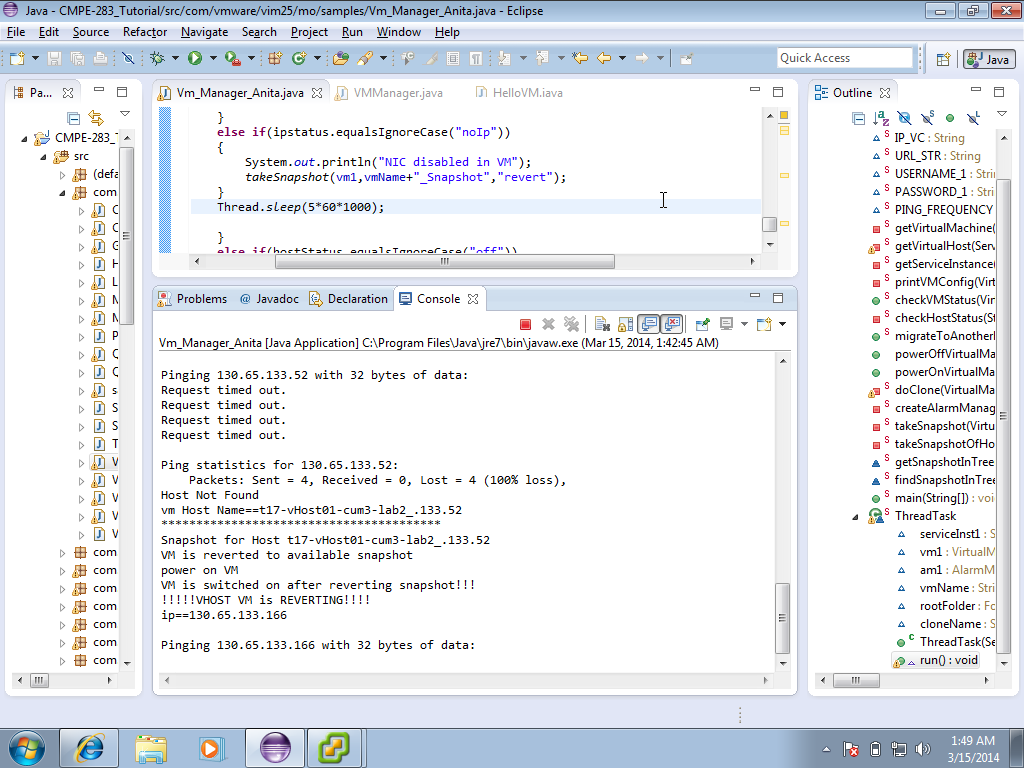
**Screenshot-5:** When the virtual machine is alive, the snapshot of the virtual machine is taken and updated in the cache every 10 minutes.

****

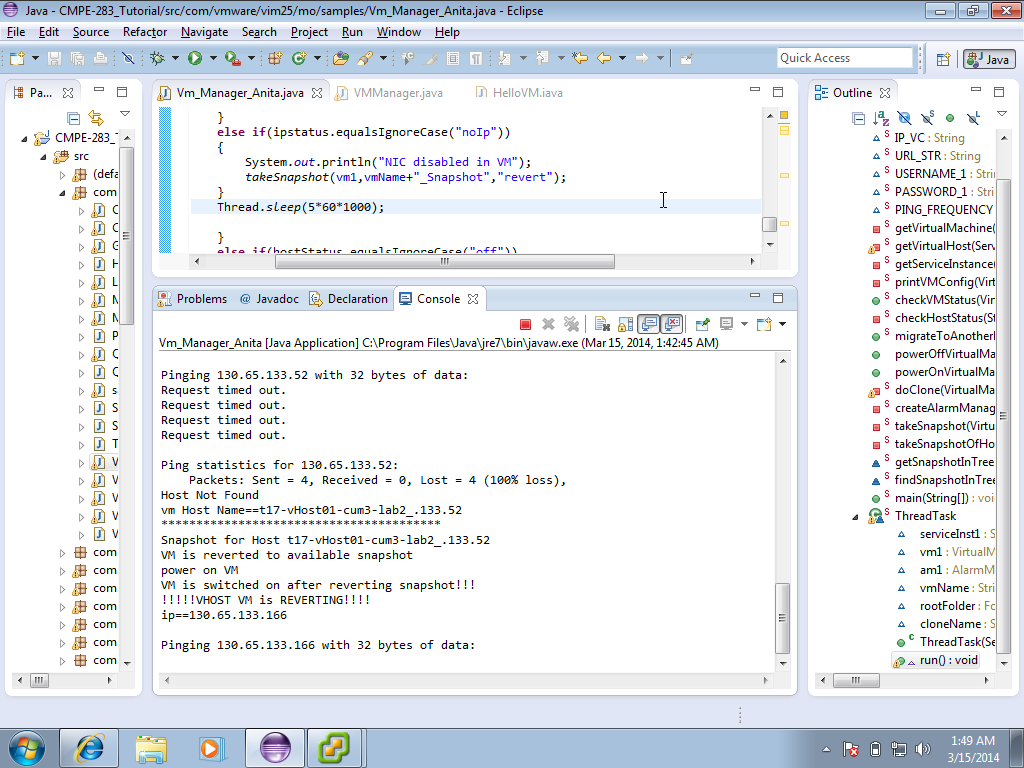
**Screenshot-6:** In order to check the failure scenario, the Network Interface Card (NIC) of the virtual Host is disabled.

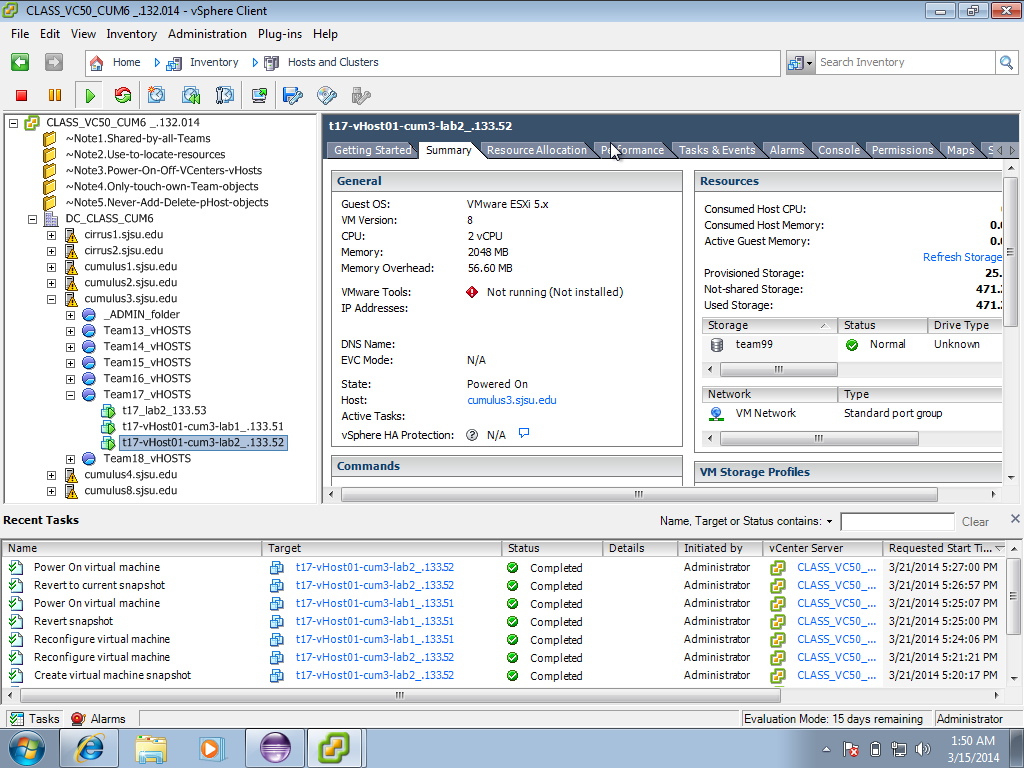


**Screenshot-7:** Once the NIC card is disabled, the vHost stops replying to ping.

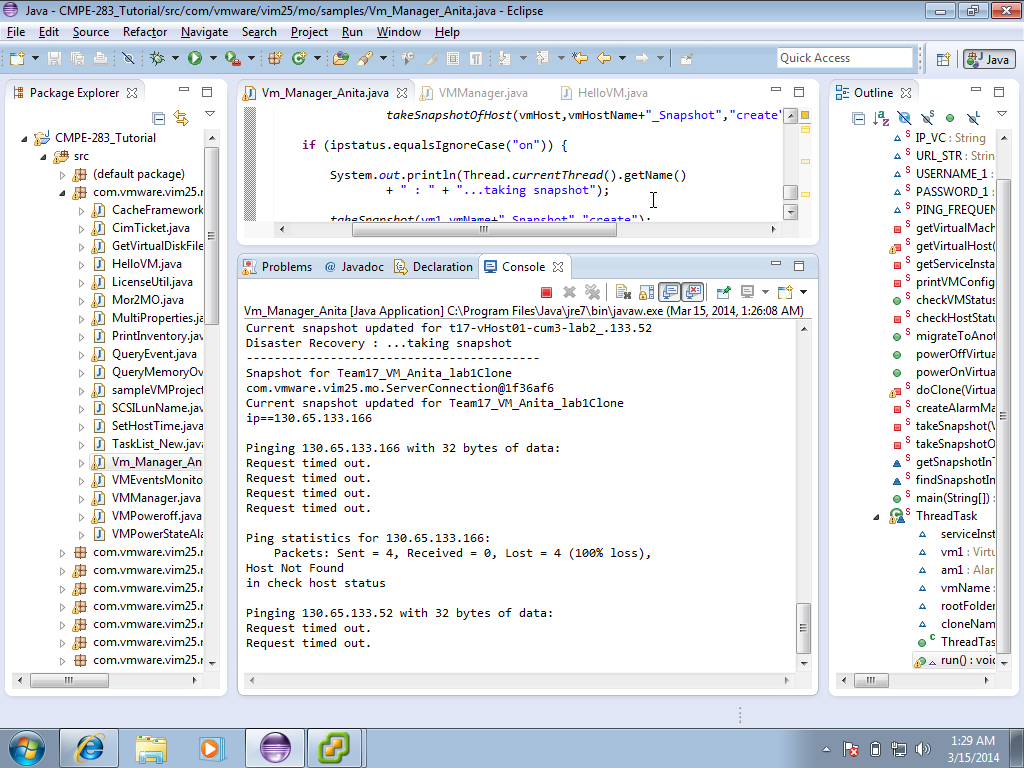
****

**Screenshot-8:** The vHost is reverted back to its previous healthy state by using the snapshot in the cache.

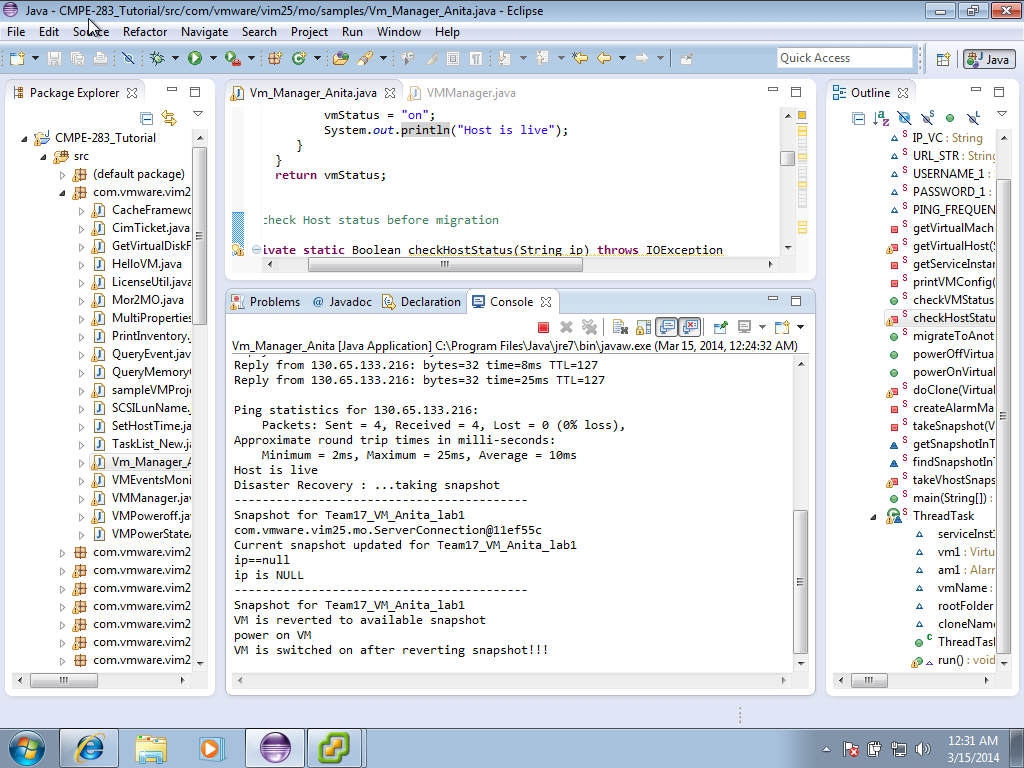


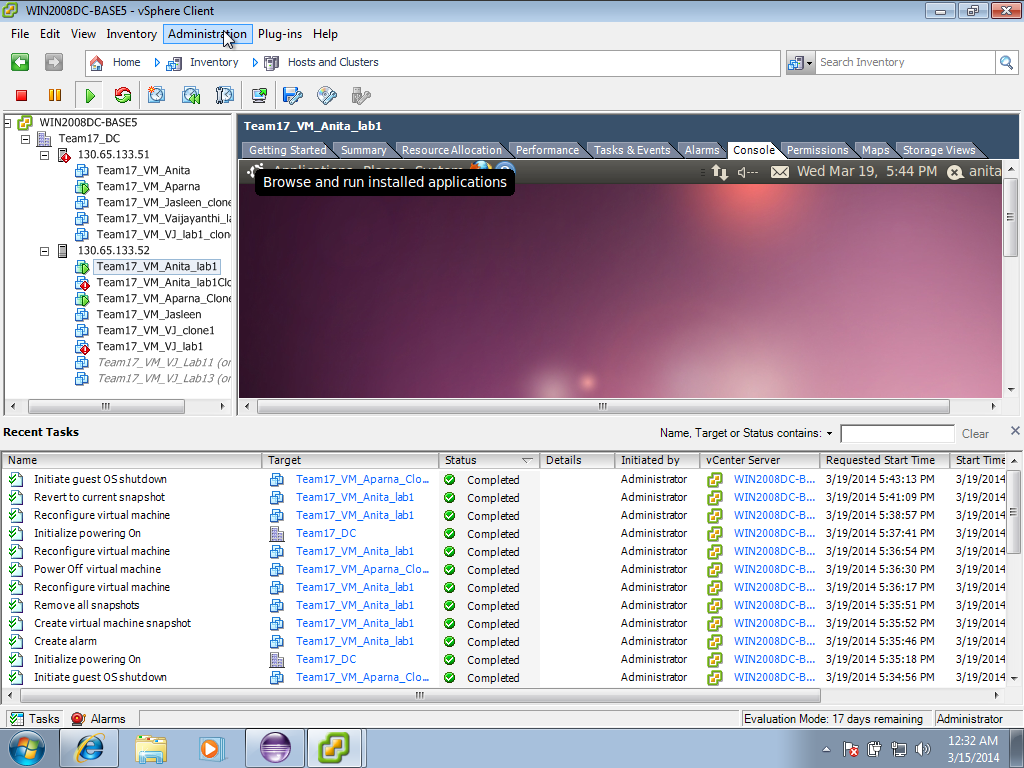


**Screenshot-9:** The similar scenario is tested in the virtual machine, by disabling the NIC of the virtual machine. Hence the virtual machine stops replying to ping.

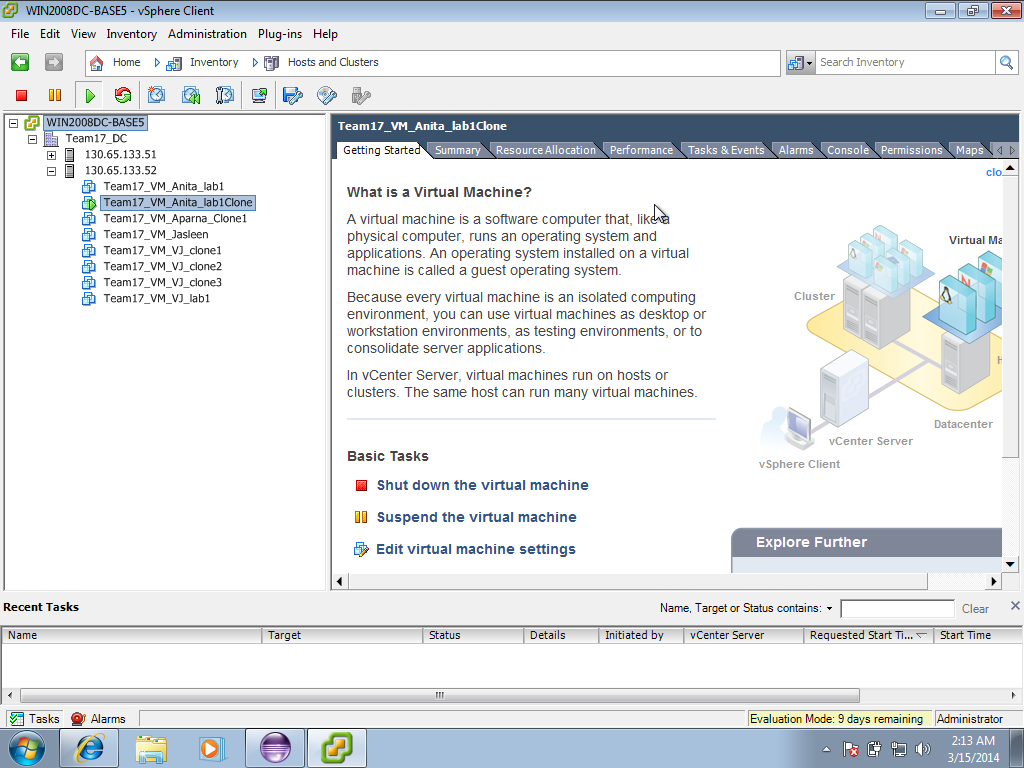


**Screenshot-10:** The virtual machine is reverted to its previous active state using the snapshot in the cache.

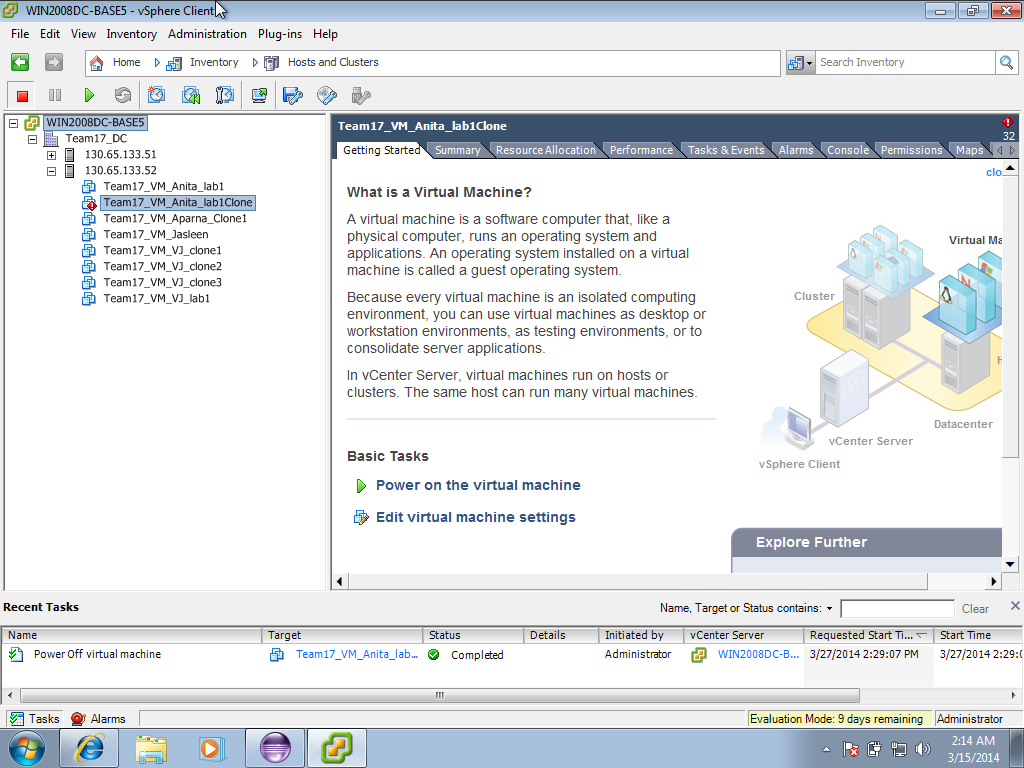
****

****

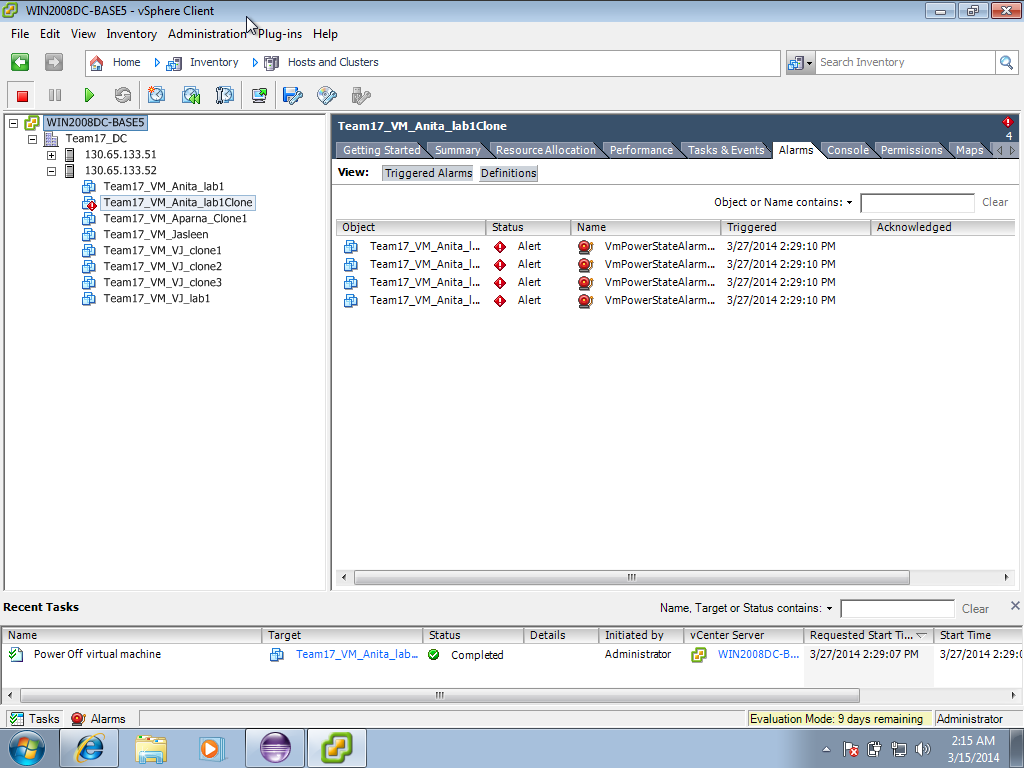
**Screenshot-11:** Virtual machines before the alarm is triggered



**Screenshot-12:** The virtual machine is powered off.



**Screenshot-13:** The alarms are triggered once the state of the virtual machine is changed.



**6. ANSWERS TO QUESTION**

**1. In case of failure, what is the good approach during Disaster Management of Virtual Machines?**

1. **Check the Host first and then the virtual machine**
2. **Check the virtual machine and then the host**

**Justify the answer with sufficient reasons.**

In times of failure of the virtual machines, the best approach that can be adhered by the disaster recovery management system, would be to check the status of the host first and then the virtual machine.

This approach is good because when the host fails, the virtual machine automatically get disconnected, becomes inaccessible and stops working. Hence its wise to check the status of the host, If the host is failed it means the virtual machine is also unavailable.

In the availability manager that is developed, the status of the host is checked first, if its alive then the virtual machine status is checked. If the virtual machine doesn’t reply to ping, the virtual machine is made alive by using its snapshot and is reverted back to its previous healthy state. If the host fails, and the host stops replying to ping, then the host is reverted back to its previous healthy state using the snapshots taken and then the status of the virtual machine is checked.

**2. How many threads do you think are required for the optimal functioning of your Availability managers? Support your answer with reasons.**

For the optimal functioning of the availability managers, two threads will be required. The first thread sets the alarm for the virtual machines to alert in case of change in the status of the virtual machines. The other thread can be used to ping the host and then the virtual machine and check their health status.

**3. How did you set and use the alarms to aid your availability manager? Where else can you use the alarms? Explain.**

Alarms are used in virtual machines to trigger notifications or alerts when specific incidences occur in the virtual machines. In the availability manager that is implemented, alarms are used to monitor the state of the virtual machine. Similarly it can be used to check for the CPU usage, when the usage exceeds a designated amount alarms can be triggered. Alarms can also be used to identify errors in the virtual machines.

**7. DISCUSSION**

In the Implementation of the availability manager, the host add/ remove mechanism is not imparted as the availability of the machines, both host and VMs can be handled by taking snapshots and reverting the snapshots in times of failure. Adding additional features, which may not be used increases the complexity and the time involved in the implementation.

The most important part of the availability manager is to find the status of the machines at regular intervals of time and detect the failures appropriately. For this the virtual machines are pinged regularly for every 10 minutes. When the machines stop replying to the ping, the availability manager assumes something is wrong and hence declares the machine is failed.

The approach, which is used to detect the failure of the virtual machine, is used to detect the host failures too. The availability manager pings the host IP periodically for every 10 minutes, when the host stops responding the same mechanism, which is used to revert the virtual machine, is used. In this way, the availability of the host and the VMs are ensured. Disabling the network interface card can simulate the failure of the hosts and virtual machines, which can be used for testing.

**8. CONCLUSION**

The following conclusions were made from the project experience:

**1. LESSONS LEARNT:**

1. This disaster recovery is rapid, reliable and cost-effective. This approach explained about the detailed working of hypervisors and their corresponding management servers. Usage of virtual environment made the disaster recovery process comparatively quick and manageable. The project focused on understanding the results of a failed host and a method to recover the failure, on different host by creating a snapshot of the previously working state of the VM.

2. Got a chance to explore the capabilities offered by these components and their APIs

3. Learnt about the issues arising from interoperability

**2. CHALLENGES:**

1. Monitoring the status of vHost and recovering the vHost to its active state.
2. When the vHost is failed, the vHost is recovered by using the snapshot taken when it was active, when vHost fails, the virtual machines running in that host will also fail and hence the virtual machines should also be reconfigured using the snapshot taken.
3. Alarms are created to monitor the status of the virtual machines, hence when ever there is a change in the state of the virtual machine, alarm is triggered.

**3. RECOMMENDATION FOR FUTURE PROJECTS:**

In the future projects more scenarios to use the alarm manager can be included. Alarms can be created for virtual hosts along with the virtual machine. If an alarm is generated that a virtual host is powered off, then pinging the virtual host can be avoided. This handling can also be imparted as part of the project.

**8. REFERENCES:**

1. <http://vijava.sourceforge.net/faq.php>

2.<http://pubs.vmware.com/vsphere55/index.jsp?topic=%2Fcom.vmware.wssdk.pg.doc%2FPG_Preface.html&__utma=207178772.1963450761.1395815707.1395815707.1395815707.1&__utmb=207178772.0.10.1395815707&__utmc=207178772&__utmx=-&__utmz=207178772.1395815707.1.1.utmcsr=google|utmccn=(organic)|utmcmd=organic|utmctr=(not%20provided)&__utmv=-&__utmk=83418133>

3. <https://www.vmware.com/pdf/vi_architecture_wp.pdf>

4.<http://pubs.vmware.com/vsphere-4-esx-vcenter/index.jsp?topic=/com.vmware.vsphere.webaccess.doc_40/managing_virtual_machines/c_viewing_information_about_virtual_machines.html>

5.<https://www.vmware.com/files/pdf/data_recovery_datasheet.pdf>