A Resource Management Methodology for Collaborative Computing System over Multiple Virtual Machines

Xiaojun Chen
School of Computer science and engineering, Xi'an University of technology, Xi'an, 710048, China,
Email:army.net@163.com

Jing Zhang , Junhuai Li and Xiang Li
School of Computer science and engineering, Xi'an University of technology, Xi'an,710048,China
Email: zhangjing@xaut.edu,lijunhuai@xaut.edu.cn, lixiang@163.com

***Abstract*—A resource management methodology for collaborative computing systems over multiple virtual machines (CCSMVM) is presented to increase the performance of computing systems by improving the resource utilization, which has constructed a scalable computing environment for resource on-demand utilization. We have designed a resource management framework and a prototype to improve resource utilization rate, reduce computing systems overheads and maintain workloads balancing, whose key technologies include resource planning, resource allocation, resource adjustment and resource release. The experiments have verified the feasibility of our prototype and the results of system evaluations show that the time of resource allocation and resource release is proportional to the quantity of virtual machines, but not the time of the virtual machines migrations. Our study on resource management methodology has some significance to the optimization of the performance in virtual computing systems.**

***Index Terms*—Virtual machine; Virtualization; Collaborative computing; Resource management; Resource utilization**

I. INTRODUCTION

Package and isolation characteristics provided by virtualization technology make the platforms that applications host isolate from the underlying environments. We no longer require to adjust applications frequently based on the changing conditions in virtual computing systems, and just need to construct a software distributed to different types of platforms packaged by the virtualization, as a result, the deployment of applications is to deploy a virtual machine with an operation system and applications to a physical machine [1-2]. Virtual machines, the logic files in virtualized data center which package corresponding VCPU and memory resources, are isolated by a management program. The executing of an application and the creating of an virtual machine can be combined by this management program which packages the guest OS and other sharewares as a virtual appliance in order to save the time of virtual machine creation. In this mode, the virtual machines are cloned and their parameters are changed before they start to run [3].The deployment of a virtual machine demands us to determine the resources required firstly, which means that the problem of resource management over multiple virtual machines is the core problem to be solved presently [4-5].In data center, users see a unified virtual computing resources pool, rather than a lot of servers isolated from each other, and virtual computing resources pool manages the CPU, memory, storage and network resources. Virtual machines can be migrated freely [6].Users could adjust the resources allocated to virtual machine timely, rather than close servers and open the installation devices again to restart the systems. The management program can dynamically adjust the resources flexibly based on current resources status inside a virtual machine through API [7] .Resource management constitutes of resource abstract, resource planning, resource allocation, resource monitoring, load management, resource deployment, resource recycle and so on [8-15]. The research hotspots of resource management in virtualized platform are virtual machines monitoring and management, virtual machines migrations and distributed resources scheduling technologies [16-22].

Collaborative computing is a hotspot topic and collaborative computing system over multiple virtual machines (CCSMVM) would enable the applications match the computers' architectures in a higher level because the virtualization would increase the peak performances of computing systems by improving their resource utilization rate[23-29]. CCSMVM completes parallel processing with nondestructive performance in a virtualized platform by using a flexible architecture with the self-adaptability and transparency [30-32]. Resource management methodology has also become a core problem of CCSMVM. Present resource management policies of virtual machines tended to be artificial configuration, which can not satisfy the resource requirements from a lot of tasks. Artificial resource

doi:10.4304/jsw.6.11.2282-2291

configuration rather than automated configuration would lead to a higher management costs and a greater workloads. It is becoming the bottleneck to improve resource utilization rate. Meanwhile, opaque virtual machines migrations policies would make it difficult to adjust the resources in tasks running. In the solution oriented the requirements of collaborative computing, virtual machines should be deployed automatically and the resources should be scheduled automatically. Based on this requirement, we take the resource management methodology in CCSMVM as the focus in this paper. A resource management methodology for resource planning, resource allocation, resource adjustment and resource release has been proposed. We developed a prototype system to implement those technologies and made some system evaluations to verify our work.

II. RESOURCE MANAGEMENT FRAMEWORK

We present a resource management framework and describe its key supporting technologies in this section.

A.Resource management framework

A resource management framework oriented *CCSMVM* , presented in Fig.1, mainly constitutes of four parts: cluster layer, server layer, client layer and user layer. Our resource management framework bases on *C/S* structure and takes *Xen hypervisor* as the underlying virtualized platform. User requirements, user applications codes and running results of tasks are submitted to client layer. Resource configuration and resource scheduling are located in server layer, and resource monitoring is implemented in cluster. We introduce their modules as follows:

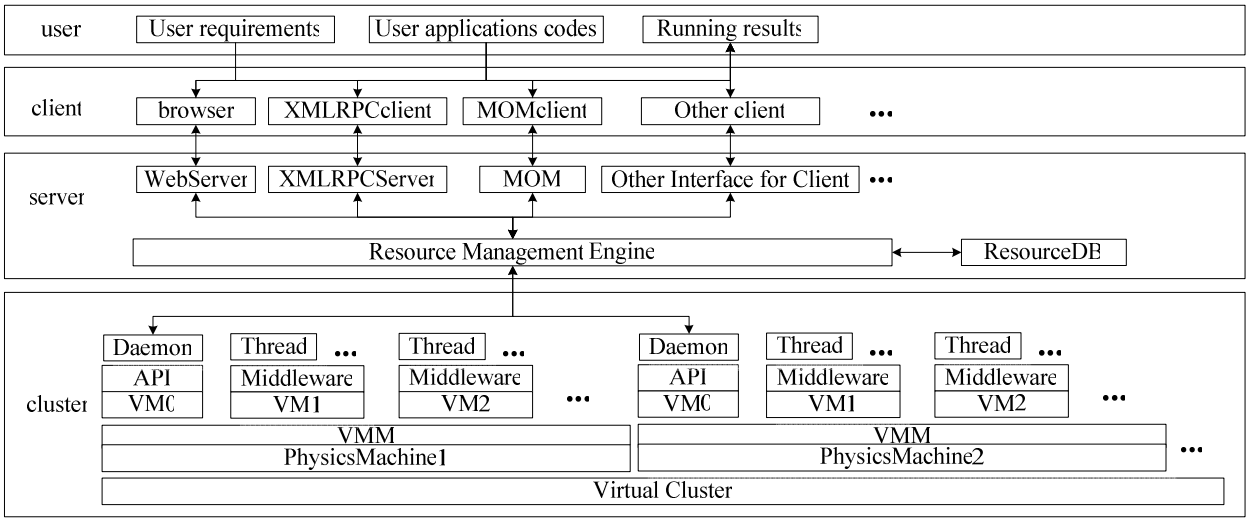

Fig.1 Resource management framework

1) ***VM*** constitutes of *VM-0* and *VM-U*. *VM-0*, the privileged virtual machine, monitors the resources in host, and *VM-U*, the virtual machine created by calling *Xen API* in the daemon of *VM-0* according to requests from *resource scheduling engine*, executes tasks exclusively.

2) ***Middleware in virtual machine*** provides a stable environment for the applications and manages the applications in virtual machines.

3) ***API***, an interface that *Xen hypervisor* provides to applications, manages the virtual machines in host.

4) ***Daemon***, a monitoring program running in *VM-0*, collects the static and dynamic information on virtualized platform by calling *Xen API*, and then monitors resources' status of virtual machines. The resources' data are submitted to server layer, and then they are dynamically adjusted by *resource scheduling engine* based on the requests from server layer.

5) ***Resource scheduling engine*** *(RSE)* completes resource virtualization and processes resources requirements from user layer, and then determines the policies for resource allocation and resource adjustment. It connects the *daemon*s in cluster layer, which collect the resources' information and implement resource allocation and adjustment policies in *daemons*.

6) ***ResourceDB***, a resource management database, stores the static and dynamic configuration data for virtual machines and the physical machines in virtualized platform.

7) ***ServerInterface***, the tools of servers such as *WebServer, XML-RPC Server, and Middleware oriented message server (MOMserver)*, etc., provides the interfaces to connect the server and the client.

8) ***ClientInterface***, the tools of clients such as *browser,XMLRPCClient, Middleware oriented message client(MOMclient)*,etc., provides the interfaces to transmit the requests from users in client to *RSE* in server ,such as *user applications codes* and *user requirements* ,and then, *RSE* submits the *running results* to users.

In above modules, *RSE*, as the core module of the resource management framework, provides a methodology for resource configuration and a interface for resource reservation to manage the resources used by virtual machines. Meanwhile, *RSE* also makes polices of data acquisition and data feedback to users, so it is the

key to implement resource virtualization and resource management in *CCSMVM*.

B.Key supporting technologies

（1）Virtual appliance technology

Our resource management framework requires us to create a virtual machine within the limited time. Firstly, the privileged virtual machine *VM-0* in host is suggested to maintain some virtual appliances. A virtual appliance is a storage entity including a file or several files storing the configuration of virtual machines, such as the disks, the memories and the CPUs' status. Virtual appliance, as an image containing the minimum virtual machine, guest *OS*, middleware and their supporting modules, are pre-installed and pre-configured. The steps of creating a virtual appliance include the creation of a virtual machine, the installation of guest *OS* and the middleware. The path of virtual appliances should be managed by *RSE* which loads and actives the virtual appliances to start the virtual machines, and then completes resource configuration through users' configuration parameters. And finally, the applications are transmitted to virtual machines.

（2）Xen API

Xen API is the focus of key supporting technologies. *Xen API*, built atop *XML-RPC*, is used by the userspace components of *Xen*. The xend daemon on *VM-0* listens for *XML-RPC* connections and then performs a number of administrative functions. The *Xen API* is used to provide a bridge between the low-level daemons and userspace applications. Most Xen management functions are performed by the xend daemon. This parses the requests from userspace tools and communicates with the *VM-0* kernel to perform management functions. The Xen daemon provides an interface inbetween the rest of the userspace tools and the kernel interfaces. It might seem at first glance that xend could be implemented entirely in kernel space. It would certainly not be difficult to expose a "perform hypercall" system call, or allow userspace applications to access the Xen device directly. To see why this is not the case, it is necessary to see what xend does beyond simply issuing *hypercalls*. Not all management functions require direct interaction with the hypervisor. Some, such as starting various virtual driver back ends, happen entirely in userspace, or with a small amount of hypervisor interaction to establish shared memory regions via the grant tables. If xend was entirely kernel-based, it would be very difficult for it to perform many of these functions.

（3）The virtualization of applications

If an application closely depends on the functions provided by an operation system, such as memory allocation, device drivers, service processes and dynamic link libraries, it may have the poor adaptability in their environment. The virtualization of applications is very necessary because the dependences between applications and underlying systems are abstracted, and the coupling relationships between applications and operation systems are removed. The virtualization of applications provides us a run-time environment in virtualized platform for applications .When we start an application, its executable files are migrated to virtualized environment from *RSE*, and then, they would run directly during some easy configurations. After an application runs over, its code would be deleted timely. In general, the applications are deployed to virtual machines in the form of a stream, so the network broadband and *QoS* are necessary to ensure the real-time, availability and the ease of use in applications migration.

III. THE KEY TECHNOLOGIES OF VIRTUALIZED RESOURCE MANAGEMENT

We first define the basic concepts, and then discuss the implementation of key technologies in this section.

A. The basic concepts

The implementation of resource management framework requires us to create an environment with resource on-demand utilization. Resource virtualization, as the premise to implement resource management, transforms the physical resources to logic resources for resource on-demand utilization. Virtualized resources can be reused to improve their utilization rate and tend to complete more tasks in unit time.

Definition 1(*Resource virtualization*): Resource modules located at the low layer provides an interface called virtual resource interface to top resource modules and the expecting environment of top resource modules is consistent with original systems. Resource virtualization makes top resource users use the virtual resource interface and the tasks run in a virtual environment. The key problem of resource virtualization is to transfer the physical resources into logic resources based on resource users. We should complete the transformation from physical view of resources to logic view of resources. Let the set of physical resources be $PRS=(PR_1,PR_2\ldots PR_n)$ and the set of logic resources be $PVS=(PV_1,PV_2\ldots PV_n)$, then, we call the mapping $PRS \rightarrow PVS$ as resource virtualization ,which can be shown in formula (1).

$$PRS \rightarrow PVS = \{Y_i PR_j \rightarrow PV_i \mid PV_i \in PVS \cap \sum PR_j = PRS\} \quad (1)$$

Where, Y_i in formula (1) means the combination of elements.

Definition2（*Resource On-demand utilization*）: In multiple virtual machines, effective resource allocation can satisfy the requirements so as to increase the computing capability of limited resources with a higher resource utilization rate, a shorter resource average waiting time and a higher resource satisfaction. Let the set of resources requirements from collaborative computing tasks be $VCRS=\{VCR_1, VCR_2\ldots VCR_n\}$, then each resource requirement would include a demand for the size of resource's granularity. If each *VCRi*∈ *VCRS* can find out a different *PVi*∈*PVS* from others in virtual

computing resources pool, the expression satisfying minimum *difference* that PV_i matches VCR_i is as follows:

$$\min difference = \sum_{i=0}^{n} Granularity(PV_i) - Granularity(VCR_i) \quad (2)$$
$$st. \forall VCR_i \in VCRS \wedge \exists PV_i \in PVS \cap Granularity(PV_i) - Granularity(VCR_i) > 0$$

We call the matching status satisfying formula (2) as resource on-demand utilization.

B.The implementation of key technologies

The key technologies of resource management in *CCSMVM* are resource planning, resource deployment, resource allocation, resource adjustment and resource release. *RSE* completes such four works according to users' applications .The flow chart of resource utilization is shown in Fig.2 and its steps are as follows:

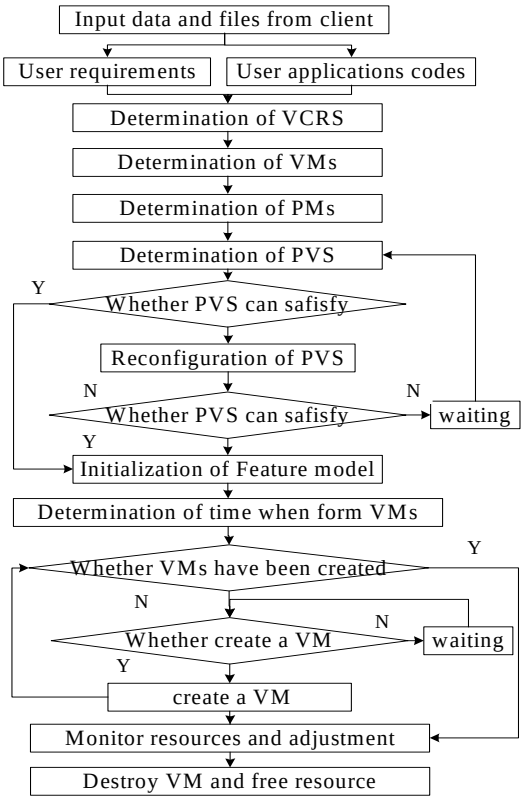

Fig.2 The resource management process

***Step1*:** Based on *user requirements*, *VCRS* are determined.

***Step2*:** The tasks are mapped into virtual machines and the virtual machines are scheduled into physical machines.

***Step3*:** *PVS* are found out to match *VCRS* to achieve resource on-demand utilization.

***Step4*:** For any *pv* $\in$ *PVS*, if *pv=null*, the virtual computing resources pool will be reconfigured according to its corresponding *VCR*. But if *pv<>null*, the flow will goto *step6*.

***Step5*:** If *pv=null*, the system reminds users that enough resources can not be obtained from virtual computing resources pool and the user needs to wait. But if *pv<>null*, the flow will goto *step6*.

***Step6*:** Based on the dependences from users and collaborative computing tasks, the creation opportunities of virtual machines are determined.

***Step7*:** Virtual machines start, and the resources included in *PVS* are allocated. Then, the virtual appliances are loaded and active, and the executable files for this application are migrated and deployed.

***Step8*:** Daemons in *VM-0* monitor the resources used by these virtual machines, and resource reconfiguration is performed to adjust the granularities of logic resources based on the change of resources.

***Step9*:** When tasks complete, the virtual machines are destroyed and the resources allocated are released.

We analyze these steps in detail as follows:

(1)Resource planning

Resource requirements should be determined based on the workloads of tasks. The quantities of virtual machines and physical machines should be determined before we distribute the executables files. We determine the virtual machines required to create based on their threads description of tasks. The determination of physical machines refers to the scheduling of virtual machines to physical machines. *Xen hypervisor* is responsible for the scheduling of virtual machines deployed in host. The steps of resource planning are as follows:

Step1: The mapping from tasks to virtual machines. The mapping from threads of tasks to virtual machines solves the problem of decomposing the tasks and deploying their subtasks into virtual machines by serial or parallel computing.

***Step2*:** The deployment of virtual machines to physical machines. Based on the quantity of physical machines in cluster, the virtual machines deployment to physical machines should keep the workload balancing.

Step3: The determination of the creating opportunities of virtual machines. Before the tasks start, the time preparing subtasks are bonded to determine the creating time of virtual machines.

Because the overheads of resource planning in virtual computing systems are greater than that in non-virtualized system, we should determine the time of virtual machines creating opportunities. If the virtual machines are created too early, the resources would be wasted. But if too late, resources can not satisfy the requirements of applications. The subtasks in non-collaborative computing tasks have no strict dependence, so the creating opportunities of virtual machines are determined directly based on requirements from resources users, but the creating opportunities of virtual machines with latter subtasks should be determined by the progresses of frontier subtasks because of their strict dependences among virtual machines. In this mode, the subtasks should access *RSE* to compute the progresses of frontier subtasks.

(2)Resource allocation

Based on the requirements of applications, *RSE* modules send out requests to daemons by calling *Xen API*, and then *VM-0* allocates the resources to create virtual machines required. The creation of virtual machines refers to the allocation of resources. We should find out the set of logic resources *PVS* matching *VCRS* firstly, and then construct the virtual machines by using resources in *PVS*. The steps of resource allocation are as follows:

***Step1*:** Find out the set of physical resources *prs* that *pv* maps to.

***Step2*:** Create and allocate VCPUs, determine the memories required, and allocate the I/O devices.

***Step3*:** Initial the virtual appliance with operation system, library, middleware and runtime environment, and then adjust the parameters of virtual appliance.

***Step4*:** Distribute the executable files and data files, and set the path of these applications.

***Step5*:** Initial the parameters of applications in virtual machines and start these applications.

(3) Resource adjustment

After virtual machines run, we adjust their resources based on the results of resources monitoring. As the resources requirements are not accurately assessed by tasks created early, the system is required to monitoring the resource utilization of CPUs, memories, disks and network devices, and then determines whether the resources granularities allocated originally match the resources granularities used or not. Resource adjustment refers to the making of virtual machines migrations policies based on resources monitoring results. And then, virtual machines migrations are completed by calling *Xen API*. The steps of resource adjustment are as follows:

***Step1*:** Obtain the actual CPU utilization that the tasks require;

***Step2*:** Obtain the actual memory sizes that the tasks require;

***Step3*:** Obtain the actual device utilization such as network and disk that the tasks require;

***Step4*:** Determine actual granularities that virtual machines required according to above resource utilization, and then determine the granularities of CPUs, memories, disks and network devices.

***Step5*:** Determine whether the virtual machine hosted in a physical machine should be dynamically migrated. If the virtual machines should run in other physical machines, then we should carry out virtual machines migrations and refresh the relationships of elements in PVS used. If not, we should update related parameters such as CPUs, memories, I/O devices for these virtual machines, and then, adjust the relationships between logic resources and physical resources.

(4) Resource release

When the tasks have been completed, *RSE* tells the daemons to delete the files of these tasks, and further destroy virtual machines to release computing resources by calling *Xen API*. And then, the resources allocated return to virtual computing resources pool. The steps of resource release are as follows:

***Step1*:** Terminate the applications executing in virtual machines and release the resources that the applications occupied.

***Step2*:** Close the executing environment of applications.

***Step3*:** Close the virtual appliances, such as guest OS, libraries, middleware and other software.

***Step4*:** Release CPUs, memories and I/O devices occupied by virtual machines.

***Step5*:** Logic resources and their physical resources are set to untreated state in *ResourceDB*.

IV. PROTOTYPE IMPLEMENTATION AND SYSTEM EVALUATIONS

Based on the theory proposed, we developed a prototype system for resource management in CCSMVM, which constitutes of three modules: *resource scheduling engine in server*, *daemon of VM-0 in host* and *resource management system in client*. The prototype system has implemented *Xen API* calling in java language. In general, *daemon of VM-0 in host* is responsible for the communication with *Xen API*. *Resource scheduling engine in server* is responsible for collecting the resources from *daemon of VM-0 in host* and then implementing resource virtualization which makes resources searching and resource utilization more convenient. *Resource management system in client*, with which collaborative computing tasks are submitted, is used by users directly to change the resources' status in an easy operation. We implement such three modules by *XML-RPC* which provides a good idea for the integration of modules installed in different locations with the uniform *XML* and platform-independent language. In such three modules, we have considered the simplicity, correctness, consistency and integrity. We stress on the simplicity in the first place because it is the most important matter in design steps. Secondly, design steps must be correct and observed because the importance of the correctness is after of simplicity. We assure that the design steps are consistent with each other and the integrity covers all cases. *XML-RPC* can satisfy above four demands, and our light-weight server developed with *org.apache.xmlrpc* in java language would waste the resources as less as possible.

A. Prototype implementation

The structure of prototype system with resource management methodology is shown in Fig.3 including 3 parts as follows:

(1) ***Resource scheduling engine in server*** *r*uns a service thread *LightweightXmlRpcServer* which has implemented a XML-RPC server. The function *registerHandlers* in *LightweightXmlRpcServer* registers three classes *Scheduler*, *DBoperation* and *MessageNotice* to this server as *Handles*. *Scheduler* is responsible for monitoring the requests of tasks from client by managing *TaskDB* and the tasks' request are submitted to class *TaskComposite* to treat with *sendTaskOperation*. *DetermineVMs* in *TaskComposite* is taken to determine and deploy the virtual machine. *DeterminePMs* is taken

to determine and deploy physical machines. *triggerResourceModule* is taken to allocate the resources; *monitoringResource* is taken to monitor the resources and adjust the resources in time. The *freeResource* is taken to release the resources. Class *DBoperation*, used to read and write the database *ResourceDB*, receives the requests from *daemon of VM-0* and r*esource management system in client.* Class *MessageNotice* is to receive the requests from *resource management system in client*, whose *setDBUrl* is to set the location of database *ResourceDB* and *testCommunication* is to test the state of *resource scheduling engine*.

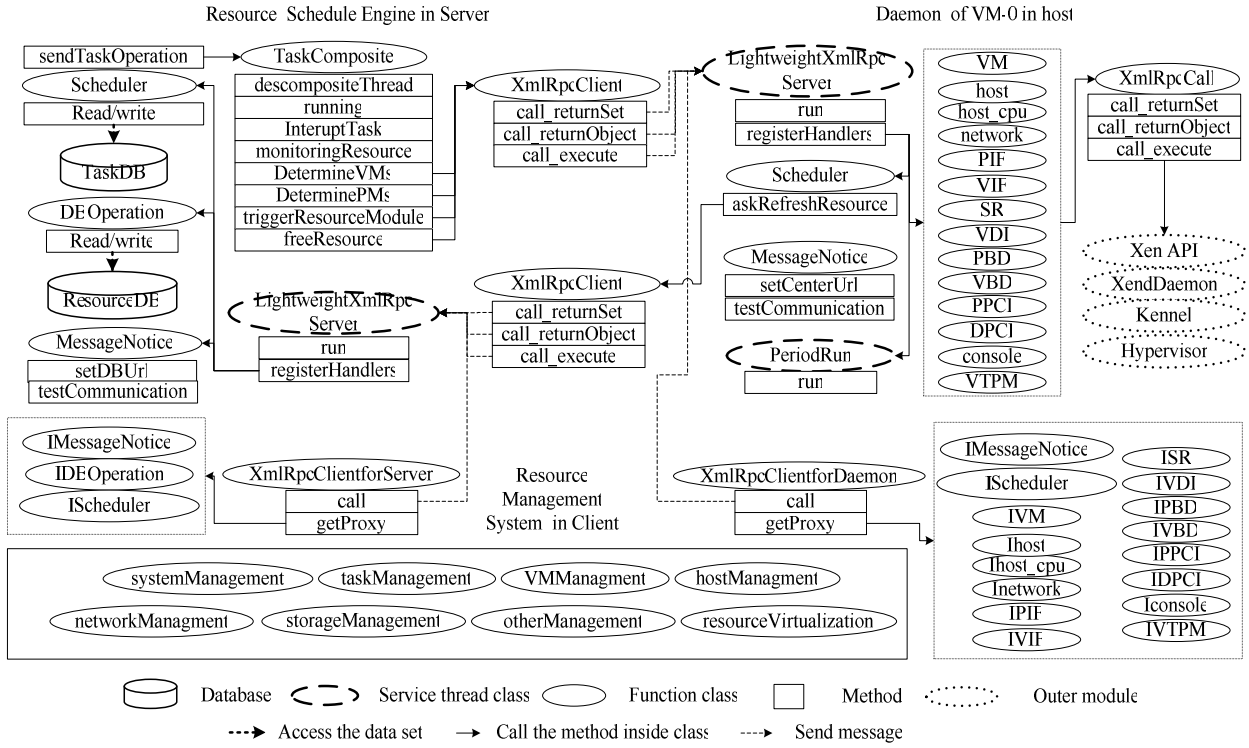

Fig.3 The workflow of resource management system

(2) ***Daemon of VM-0 in host*** runs two service-threads: *LightweightXmlRpcServer* and *PeriodRun*. *LightweightXmlRpcServer* is a *XML-RPC* server whose function *registerHandlers* registers classes *Scheduler*, *MessageNotice* and other 14 classes such as VM, *etc.* to implement the core services. *Scheduler* is taken to complete the resources' requests from *resource scheduling engine* and *resource management system in client*, implemented by calling *askRefreshResource*. *MessageNotice* is taken to receive the requests from *resource management system in client*, whose *setCenterUrl* is to set the location of *resource scheduling engine* and *testCommunication* is to test the state of *daemon of VM-0*. Fourteen classes for resource disposition achieve the key classes of *XenAPI*. *PeriodRun*, extended from the class *TimerTask* in java, collects the resources information and then refreshes database *resourceDB* in time.

(3) ***Resource management system in client*** provides the interfaces of *XmlRpcClientforServer* calling *resource scheduling engine*, the interfaces of *XMLRpcClientfor-Daemon* calling Daemon of VM-0, and interfaces for displaying the running results in client. The *getProxy* in *XmlRpcClientforServer* is to obtain three core interfaces *IMessageNotice*, *IDBOperation* and *IScheduler* from *resource scheduling engine*. The *getProxy* in *XMLRpcClientforDaemon* is to obtain two core interfaces *IMessageNotice* and *IScheduler* from *daemon of VM-0*. And the interfaces for above fourteen resource disposition interfaces such as *IVM* ect. can be divided into 8 aspects shown in Fig.3.

Resource management methodology over multiple virtual machines developed in this paper not only has automated resource adjustment ability, but also has the ability of changing the resources state by artificial operation in client, so it is more comprehensive in functions compared with related work. We create a environment for CCSMVM, in which, ten hosts are taken to as the computing nodes and anther host is used as their management node. Furthermore, we use a PC machine in remote place as the user client. We install *Xen3.4* as the *hypervisor* in ten hosts with four *Intel Xeon 3.40GHz processors, 4GB RAM* and *160GB SCSI hard disk* in each node. Each processor brings a *16KB* cache and *1024KB* secondary cache. Para-virtualization guest OS *VZLinux-Xen-U 2.6.18* is used as the virtual appliance whose storage size is *2.1G* in files maintained in *Xen-0* for other computing nodes. *Resource management system in client* can run in *Windows XP*, *Windows vista* or *Windows 7*. *Resource scheduling engine* can run in *Windows server 2003* or *Windows server 2008*, which takes *SQL server2005* as the *DBMS* in management node. All hosts are connected by high-speed network and they form a cluster system.

B. System evaluations

The prototype we developed emphasizes on a task as a service. In this section, we take an example to analyze its resource management methodology, and then further determine the performance of resource utilization including resource allocation, resource release and resource adjustment. We take the simulation of cold flow impulsive experiment for a car engine *(CFIE*) as an instance, which is the typical coupling process considering the affecting relations between flow field and structure. *CFIE*, as a collaborative computing task, constitutes of 59 subtasks after we determine its requirement. We set the numbers to them and determine their timing-relationships, and then, submit a task of CFIE to resource scheduling engine from the client.

To determine the performance of our prototype system, we use 7~19 virtual machines to run the task of *CFIE* in 4~7 physical machines. The aims of experiment are to determine the time of resource allocation in the creation of virtual machines with the increase of the quantity of virtual machines deployed, the time of resource recycling in destroying virtual machines with the increase of the quantity of virtual machines released, and the time of virtual machines migrations with the increase of the quantity of virtual machines migrated. We run our *CFIE* programs for 10 times and then compute their average values from the results. We draw the curves by using our experiment results and make some comparative analysis:

The trend of resource allocation time with the increase of the quantity of virtual machines is shown in Fig.4 under the conditions of different quantities of physical machines. We make our experiments in *7~19* virtual machines based on *4~7* physical machines. It is seen from the experiment that resource allocation time presents the trend of increase with the increase of the quantity of virtual machines and it is proportional to the quantity of virtual machines. The more the quantities of physical machines are, the faster the resource allocation speeds are, because the parallel resources allocation of virtual machines in multiple physical machines would lead to a fact that the more the quantities of physical machines, the higher the parallelism is, and the faster the resources allocation speeds are. In our hardware environment, the resource allocation time for a virtual machine hosted in a physical machine is about *5~7* seconds.

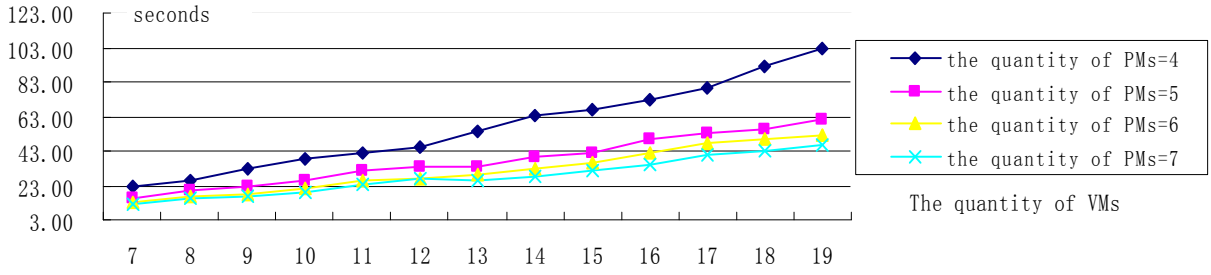

Fig.4 The trend of resource allocation time with the quantity of VMs

The trend of resource release time with the increase of the quantity of virtual machines is shown in Fig.5 under the condition of different quantities of physical machines. We release the resources allocated to virtual machines created before. It is seen from the experiment that the resource release time is proportional to the quantity of virtual machines because the virtual machines are evenly deployed to physical machines in frontier resource allocation. The more the quantities of physical machines are, the faster the resources release speeds are. In our hardware environment, the resource release time for a virtual machine in a physical machine is about 2~4 seconds.

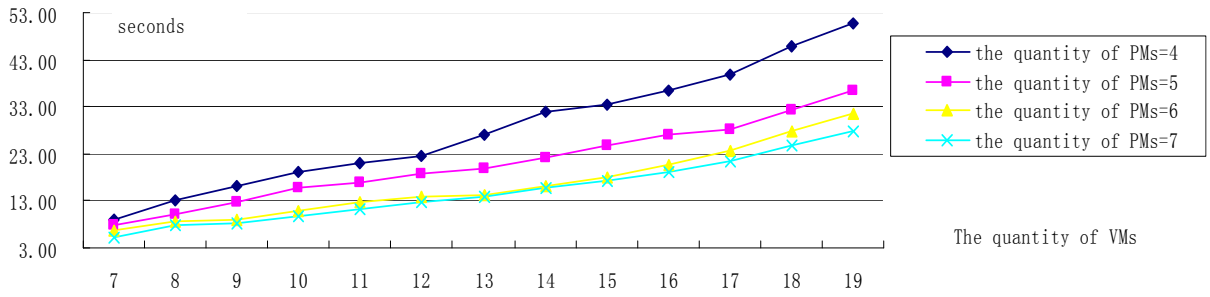

Fig.5 The trend of resource release time with the quantity of VMs

The trend of resource adjustment time with the increase of migration quantity of virtual machines is shown in Fig.6 under the condition of different quantities of physical machines. We migrate the virtual machines with the quantities from 1 to 15 based on 4~7 physical machines. It is seen from the experiment that there are the

same resource adjustment time for different physical machines if the quantities of virtual machines migrated are less than or equal to that of physical machines. We call the quantity of physical machines with the same quantity of virtual machines as the critical point, and only when the quantity of migrated virtual machines increases to this critical point, can the resource adjustment time present the trend of increase. And in this time, the more the quantities of physical machines are, the faster the resource adjustments speeds are. Because a few virtual machines would lead to a higher parallelism, the speeds of the migration become more faster. When the quantities of physical machines arrive to their critical point, the speeds of virtual machines migrations are necessity to decrease. It is shown from Fig.6 that if the virtual machines migrations arrive to critical point, the more the quantities of physical machines are, the more the quantities of virtual machines are. In our hardware environment, the resource migration time for a virtual machine hosted in a physical machine is about 1~2 seconds.

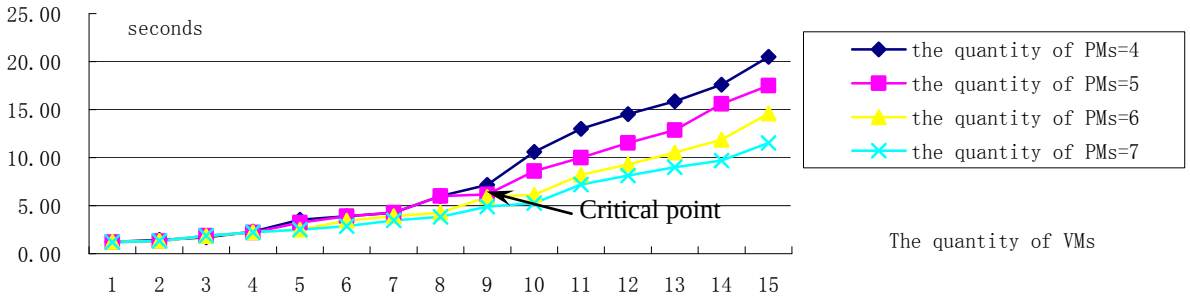

Fig.6 The trend of resource adjustment time with the quantity of VMs

It is seen from three experiments of *CFIE* that the prototype we developed has a higher efficiency in performance compared with related researches and it has achieves the goal of dynamic resource management because of its good integrative manifestation.

V. CONCLUSIONS

A resource management methodology in collaborative computing over multiple virtual machines is presented to increase the peak performance of computing systems and achieve a scalable computing environment centered on tasks. Resource virtualization, as an environment for resource on-demand utilization completes the mapping from physical resources to logic resources. We designed a resource management framework in CCSMVM based on *Xen hypervisor*, whose core modules are *daemon of VM-0* and *resource scheduling engine*. The key supporting technologies are virtual appliance technology, *Xen API* based on *XML-RPC* and the virtualization of applications. We study the key technologies of resource management, such as resource planning, resource allocation, resource adjustment and resource release. We developed a prototype for resource management methodology in CCSMVM, which provided a light weight server of multi-threads services based on *XML-RPC* with a fewer resources occupation and a higher throughput than related systemes, and it is beneficial to the integration of heterogeneous components. We made the experiments of system evaluations in our prototype and the results show that the time of resource allocation and resource release is proportional to the quantity of virtual machines, but not the time of the virtual machines migrations, because it has a critical point in its trend of increase. It is proved that the prototype has a good performance and comprehensive integrative manifestation. Our prototype had achieved the goal of resource management, so it is of good popularization value. The future work is to further improve the performance and raise the efficiency of our prototype.

ACKNOWLEDGMENT

We thank the State 863 projects of China 2007AA010305) and the excellent doctor degree dissertation fund of Xi'an University of technology (102-211007) to support this research.

REFERENCES

[1] H.Jin, X.F Liao, "Virtualization Technology for Computing System", *China Basic Science .China*, vol.10 , pp.12-18,June 2008.

[2] P.Barham , B. Dragovic, K. Fraser, et al, "Xen and the Art of Virtualization", *Proc.19 of the ACM Symp.on Operating Systems Principles , New York, USA* ,2003,pp.164-177.

[3] K.A.Fraser, M. H.Steven, M. Ian Leslie et al. "The XenoServer Computing Infrastructure", *Technical Report UCAM-CL-TR-552, University of Cambridge,U.K*, 2003.

[4] N. Michael, B.H. Lim, and G. Hutchins. "Fast Transparent Migration for Virtual Machines", *Proceedings of USENIX'05, Anaheim, CA*, 2005, pp.25-25.

[5] A.A.Waldspurger. "Memory Resource Management in VMware ESX Server", *The Symposium on Operating Systems Design and Implementation, PaloAlto ,CA*, 2002, pp.181-194.

[6] B.Sotomayor, K.Keahey, I. Foster, "Overhead Matters: A Model for Virtual Resource Management", *First International Workshop on Virtualization Technology in Distributed Computing, Washington, DC, USA*, 2006, pp. 5 - 6.

[7] L. Grit, D.Irwin, A.Yumerefendi, el al. "Virtual Machine Hosting for Networked Clusters: Building the Foundations for Autonomic Orchestration", *First International Workshop*

on Virtualization Technology in Distributed Computing,. Tampa, FL, 2006, pp. 5-7.
[8] B.K. hargharia, S.Hariri,. M.S Yousif,. "Autonomic power and performance management for computing systems", *Journal Cluster Computing*,vol.11, pp. 145-154 ,June 2008.
[9] K. Begnum, N.A. Lartey and X.Lu, "Simplified cloud-oriented virtual machine management with MLN", *The Journal of Supercomputing*, vol.5931/2009, pp.266-277, March 2009.
[10] S.Samar, K.S.Tripathi, "A Time-Slotted-CDMA Architecture and Adaptive Resource Allocation for Connections with Diverse QoS Guarantees", *Wireless Networks archive*, vol.9, pp. 479 – 494, May 2003.
[11] L.Lavy, O.Ariel, "Atomic Resource Sharing in Noncooperative Networks", *Telecommunication Systems*, vol.17,pp.385-409, April 2010.
[12] D.B.Tracy, J. S. Howard, A.Anthony, et al, "Static resource allocation for heterogeneous computing environments with tasks having dependencies, priorities, deadlines, and multiple versions", *Journal of Parallel and Distributed Computing* ,vol.68, pp. 1504-1516,Nov 2008.
[13] A. M. Michael, L.Abraham, F.Michael, et al ,"Autonomic Clouds on the Grid", *Journal of Grid Computing*, vol.8, pp.1-18, Jan 2010.
[14] G. Monia, G.Sudhakar and Gholamali C. S, "Resource optimization algorithms for virtual private networks using the hose model", *Computer Networks* ,vol. 52, pp.*3130*-3147,Nov 2008 .
[15] K.Yomi, I. Gülfem and B.B.Ayşe , "Resource allocation in cellular networks based on marketing preferences", *Wireless Networks*,vol.16, pp.27-38, Jan 2010.
[16] D.Q Zou, S.X Du, W. Zheng, et al, "Building Automated Trust Negotiation architecture in virtual computing environment", *The Journal of Supercomputing*, vol.55, pp.69-85, Nov 2009.
[17] M.Tridib, B.Ayan, V.Georgios,et al, "Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers", *Computer Network*s,2009, vol.53, pp.2888-2904,Dec 2009.
[18] G.Daniel, R.Jerry, C.Ludmila, et al, "Resource pool management- Reactive versus proactive or let's be friends", *Computer Networks*, vol.53, pp. 2905-2922, Aug 2009.
[19] K.Sanjay, T.Vanish, K.Vibhore, et al, "Loosely coupled coordinated management in virtualized data centers", *Cluster Computing*, unpublished.
[20] A.Jussara, A.Virgílio, A.Danilo, et al, "Joint admission control and resource allocation in virtualized servers", *Journal of Parallel and Distributed Computing*, vol.70, pp.344-362, April 2010.
[21] K.Attila, K. Gabor, B. Ivona, "An SLA-based Resource Virtualization Approach for On-demand Service Provision", *Proceedings of the 3rd international workshop on Virtualization technologies in distributed computing, New York, USA* , 2009, pp. 27-34.
[22] Q.Benjamin, N. Vincent and C.Franck, "Scalability Comparison of Four Host Virtualization Tools", *Journal of Grid Computing* ,vol.5, pp.83-98, Sept 2006.
[23] K.Amit, Y.Xin, and K.L.David. "An MPI prototype for compiled communication on Ethernet switched clusters", .*Journal of Parallel and Distributed Computing*, vol.65, pp.1123-1133, Jan 2011.
[24] A.Rocco, D.M.Beniamino, M. Nicola, et al. "A hierarchical distributed-shared memory parallel Branch&Bound application with PVM and OpenMP for multiprocessor clusters", *Parallel Computing*, vol.31, pp. 1034-1047, Dec 2005.
[25] M.R.J.Qureshi, S.A.Hussain, "A reusable software component-based development process model", *Advances in Engineering Software*,vol.39, pp. 88-94,Feb 2008.
[26] S.X. Jing, F.Z. He, S.H Han, et al, "A method for topological entity correspondence in a replicated collaborative CAD system", *Computers in Industry*, vol.7, pp. 467-475, Sept 2009.
[27] S.J. Zasada, P.V. Coveney, "Virtualizing access to scientific applications with the Application Hosting Environment", *Computer Physics Communications*, vol.180, pp. 2513-2525, Dec 2009.
[28] H.B. Wang, J.Z. Huang, Y.Z. Qu, et al, "Web services: problems and future directions", *Web Semantics: Science, Services and Agents on the World Wide Web*, vol.1, pp. 309-320,April 2004.
[29] T. Muntean, "A generic multi virtual machines architecture for distributed parallel operating systems design", Proceedings of *Heterogeneous Computing Workshop 94*, *Cancun* , *Mexico*, pp. 103-109, April 1994.
[30] Q. Duan, Z.H. Liang, H.Z, et al, "Developing distributed virtual machines for the tri-integration-pattern based platform", *Proceeding of SOSE '05 Proceedings of the IEEE International Workshop ,Washington, DC, USA* , 2005, pp.143-148.
[31] V.Chaudhary, C. Minsuk, J.P. Walters, et al , "Comparison of Virtualization Technologies for HPC", *Proceeding of 22nd International Conference on Advanced Information Networking and Applications, Okinawa*, 2008, pp.861-868,2008.
[32] T.Giovanni, B.Raffaele, F.Alessandro, et al, "Dynamic Hybrid Modeling of the Vertical Z Axis in a High-Speed Machining Center: Towards Virtual Machining", *J. Manuf. Sci. Eng*, vol. 780, pp.323-331,Aug 2007.

XiaoJun Chen, male, doctoral students. He was born in GuiLin city, GuangXi province, P.R. China in September, 1980. He received bachelor degree in Department of Industrial Engineering of Xi'an University of Technology in 2004 and the master degree in School of Economics and Management of Xi'an University of technology in 2009.He entered into School of Computer Science and Engineering of Xi'an University of Technology to learn about distributed computing technology in 20006.

He engaged in soft development job from September 2004 to September 2006 in Delta. Software Company. He has published 8 papers since 2006, and the representatives are as follows: Rsource Allocation and Adjustment with Genetic Algorithm in Virtual Computing Systems, ICIC Express Letters,2010;Empirical research on quality supervision elements of cooperative manufacturing process in virtual enterprise, ICPOM, 2008 ;Research on quality supervision model for cooperative manufacturing preparing process in virtual enterprise, ICRMEM,2008;et al . Now he is engaged in the researches of virtual technology and cloud computing.

Mr. XiaoJun Chen is now a member of *IEEE* Computer Society. E-mail: army.net@163.com.

Jing Zhang, male, doctor, professor, doctoral supervisor. He was born in BaoJi city, ShaanXi province, P.R.China in November, 1952. He received bachelor degree in Department of Automatic Control of Xi'an University of Technology in 1981, the master degree in Department of Software and Theory of Xi'an JiaoTong University in 1986, and the doctor degree in Department of Systems Engineering of Xi'an JiaoTong University in 1994.

He has worked in Department of Computer of Xi'an University of Technology since 1977, and now is a professor and the PhD supervisor of School of Computer Science and Engineering, Xi'an University of Technology, in which, he worked in Computer Training Center of Ministry of Education, P.R. China in 1982 for a short time. He has published 60 papers and hosted 20 research projects in recently 10 years, of which, completed 4 State 863 projects. He has published 4 books, of which, representatives are Artificial intelligence basis(Electronic Industry Press ,BeiJing,2000), Practical Course on Computer Network (Electronic Industry Press ,BeiJing,2007) and Computer Network(Xidian University Press,Xi'an,ShaanXi, 2007). Recently he concentrates on the researches of distributed system, virtualization , grid computing and cloud computing..

Dr. Jing Zhang is now one of the national key new product projects consultants and a Xi'an information technology consultant. He is also a ShaanXi manufacturing Informatization expert, the member of Services Computing Professional Committee in China Computer Federation, member of E-Government and Office Automation Committee in China Computer Federation, the evaluation expert of National Natural Science Foundation and the member of IBM Software Innovation Center Expert Committee. E-mail: zhangjing@xaut.edu.cn.

JunHuai Li, male, doctor, professor, master supervisor. He was born in BaoJi city, ShaanXi province, P.R.China in November, 1969. He received bachelor degree in Department of Computer Science and Technology of Xi'an University of Technology in 1994 , he master degree in Department of Computer Application of Xi'an University of Technology in 1997, and the doctoral degree of Department of Software and Theory of NPU in 2002. He made a cooperation research in University of Tsukuba, Japan in 2004.

He has worked in Department of Computer Science and Engineering of Xi'an University of Technology since 1997, and now is a professor and a master supervisor of School of Computer Science and Engineering, Xi'an University of Technology. He is also the dean of Department of Computer Science and Technology and the dean of Network and Information Management Center. He has published 40 papers and hosted 12 research projects in recently 10 years, of which, completed 4 State 863 projects. He has published 3 books, of which, representatives are Practical Course on Computer Network(Electronic Industry Press ,BeiJing, 2007) and Computer Network (Xidian University Press,Xi'an,Shaanxi, 2007) and Network Security Technology (Xidian University Press, Xi'an,Shaanxi, 2010).He is engaged in the researches of network computing, distributed computing internet technology, RFID technology and web data mining.

Dr JunHuai Li is now an member of Chinese Computer Society and also an member of IEEE, E-mail: lijunhuai@xaut.edu.cn.

Xiang Li, female, doctoral students. She was born in TongChuan city, ShaanXi province in December, 1976. She received bachelor degree in Department of Computer Application of Xi'an JiaoTong University in 2000 and the master degree in Department of Software and Theory of Xi'an JiaoTong University in 2003. She has entered into School of Computer Science and Engineering of Xi'an University of Technology to learn about distributed computing technology in 2008.

After graduation from Xi'an JiaoTong university, she has been working in Shan'Xi University of Science and Technology, and now she is a lecturer of School of Electronic Engineering, ShaanXi University of Science and Technology. She has publiced 10 papers since 2000, and now she is engaged in the researches of distributed system and parallel computing with multi-cores. E-mail: lixiang@163.com.