CIVIC: a Hypervisor based Virtual Computing Environment

Jinpeng Huai, Qin Li, and Chunming Hu
School of Computer Science and Engineering, Beihang University, Beijing, China
huaijp@buaa.edu.cn, liqin@act.buaa.edu.cn, hucm@act.buaa.edu.cn

Abstract

The purpose of virtual computing environment is to improve resource utilization by providing a unified integrated operating platform for users and applications based on aggregation of heterogeneous and autonomous resources. With the rapid development in recent years, hypervisor technologies have become mature and comprehensive with four features, including transparency, isolation, encapsulation and manageability. In this paper, a hypervisor based computing infrastructure, named CIVIC, is proposed. Compared with existing approaches, CIVIC may benefit in several ways. It offers separated and isolated computing environment for end users, and realizes hardware and software consolidation and centralized management. Beside this, CIVIC provides a transparent view to upper layer applications, by hiding the dynamicity, distribution and heterogeneousness of underlying resources. Performance of the infrastructure is evaluated by an initial deployment and experiment. The result shows that CIVIC can facilitate installation, configuration and deployment of network-oriented applications.

Key words: Distributed Systems, Hypervisor, Virtual Machine, Virtual Computing Environment, CIVIC

1. Introduction

Along with the rapid development of computer and Internet technology, the network has brought together various computing resources, data resources, software resources and services resources, but the utilization of these resources is far from their full potential. However, the internet-oriented applications demand for the large-scale computing, massive data processing and global information service. At this point, how to organize and manage distributed, heterogeneous and autonomic resources, to share and coordinate resources across the autonomous domains, and finally to improve utilization of resource becomes a key issue to design the software infrastructures for network-based large scale computing systems.

One of the possible way to solve this problem is to build a virtual computing environment, on the open infrastructure of the Internet, providing harmonious, transparent and integrated services for end-users and applications[1]. Many interesting computing paradigm has been proposed from both academic and industrial communities, such as the Grid, Peer-to-Peer (P2P) systems, ubiquitous computing, and desktop computing[2-6]. Grid Computing is mainly focusing on the dynamic and across-domain network resource share and coordination; P2P is commonly used for end-to-end resource and information services; Ubiquitous computing is aiming at accessing resources at anytime and anywhere; and the desktop computing is trying to integrate large numbers of idle resources to provide computing intensive applications with high-performance computing power. All of these approaches are trying to provide a unified resource virtualization environment to get better utilization of resource capacities.

In recent years, the virtual machine technology has got wider attention and developed rapidly. Hypervisor or virtual machine[7] is a virtual layer between the hardware and software. It can provide applications with independent runtime environment by shielding the dynamic, distributed and heterogenic hardware resources. It can assign individual users an independent and isolated computing environment. And it can help system administrators to manage hardware and software resource centrally. According to these advantages, the virtual computing environment based on virtual machine has now become a hot spot, many virtual machine based mechanisms and systems has been proposed, such as Virtual Workspaces[8], VIOLIN[9], and Virtuoso[10]. However, the research of virtual computing environment based on virtual machine is still in its infancy. Existing approaches are usually focusing on single virtual machine, and lacked of the crossover of a variety of related technologies. In addition, management and monitoring features are not

2007 International Conference on Parallel Processing Workshops (ICPPW 2007)
0-7695-2934-8/07 $25.00 © 2007 IEEE

well supported in these systems, the status and the characteristics of the virtual computing environment.

In this paper, a hypervisor based computing infrastructure, named CIVIC, is proposed. Compared with existing approaches, CIVIC may benefit in several ways. It offers separated and isolated computing environment for end users, and realizes hardware and software consolidation and centralized management. Beside this, CIVIC provides a transparent view to upper layer applications, by hiding the dynamicity, distribution and heterogeneousness of underlying resources. Performance of the infrastructure is evaluated by an initial deployment and experiment. The result shows that CIVIC can facilitate installation, configuration and deployment of network-oriented applications.

This paper is organized as follows. In Section 2, a brief survey on the hypervisor technology is given by carefully designed taxonomy. In Section 3, several hypervisor based virtual computing systems are introduced. A hypervisor based software infrastructure for virtual computing, named CIVIC, is proposed in Section 4 with layered architecture and modules. Finally, the performance of the infrastructure is evaluated by an initial deployment and experiment in Section 5.

2. Virtual Machine and Hypervisor

The concept of virtual machine (VM) is still not clearly defined. When people are talking about *virtual machine*, sometimes, they refer to a type of software which can emulate a physical machine, like *Virtual PC* or *VMWare*. In this case, it is more accurate to use the word *hypervisor*, or *Virtual Machine Monitor* (VMM). Beside this, virtual machine can also refer to an instance of emulated (or virtual) machine with software installed inside. In this case, the word *virtual machine instance* is more appropriate. In this paper, we will follow this rule to avoid ambiguity.

2.1 Background

Hypervisor has a long history since it in its infancy in early 1960s. At that time, operating system is not yet mature, with the software running exclusively on hardware platforms. Sharing hardware resources was an urgent requirement for operating system designers. In 1966, VM/360 (Virtual Machine) system[11] came into being in IBM Cambridge research center VM/360, which is later treated as the first system to achieve multi-user support for computers, different applications and users can share hardware resources for different use sessions.

With the increasing complexity of application and hardware, the requirement of software reuse and portability increased simultaneously. Hypervisor technology kept growing at the same time. In 1980s, *Smalltalk 80* was proposed, which introduced programming language virtual machine, and provide the software portability. Application written in *Smalltalk* is first compiled into an intermediate machine language, then emulates by Smalltalk virtual machine at run-time. With the help of this type of virtual machine, application becomes platform independent, and portable among heterogeneous hardware platforms. Smalltalk provides partial solution to the software reuse problem, which implements "compile once, run everywhere". Today, *SUN* shares the same idea in Java language: the most commonly used programming language in Internet age.

With the popularization of personal computers, a variety of non-compatible computer platforms emerged at the same time, each with a wide-range of applications which can only be able to run on the specific platforms, such as Apple's Macintosh, and IBM's PC. It introduces the new problem of software portability and reuse. Since the Smalltalk VM can only provide portability and platform-independent features for applications written in Smalltalk, therefore does not support legacy software. However, situation has been changed in 1997, since *Connectix* provided *Virtual PC*[12] which is able to run PC applications in a Mac box without rewriting or recompiling the source code of the application. Using hypervisor technologies like *Virtual PC*, transparent portability for legacy software can be archived.

It should be noted that various early hypervisor technologies for PC platform are arming to solve the software portability problem, while ignoring the fact that hypervisor may also support the hardware resource sharing and isolate the runtime environment for software. In 1998, Stanford University set up *VMWare* Company and released *VMware* hypervisor. *VMware* is the first hypervisor software which is able to emulate virtualized PC on PC platform[13]. Although it cannot make software portable on different hardware platforms, it can provide the capability of sharing hardware and isolating software, which is a crucial requirement in many cases. It is now highly valued and widely deployed in enterprises.

Hardware may encounter many expected and unexpected situations like upgrading, maintenance and failures, which affects the reliability of software. It has become an important issue to eliminate the impact of hardware resources to enhance the stability and reliability of software. Around 2004, *VMWare* and *Xen* both proposed a kind of on-line migration technology in their own hypervisors, named *VMotion* [14] and Live

2007 International Conference on Parallel Processing Workshops (ICPPW 2007)
0-7695-2934-8/07 $25.00 © 2007 IEEE

Migration[15] respectfully. The whole computing environment based on a virtual machine can migrate from one physical machine to another physical machine without having to stop and restart the environment. more importantly, the service downtime (unavailable time) during migration can be reduced to 100ms[15]. Live migration of hypervisors can be completely transparent to software. It can be used as a dynamic load balancing mechanism for hardware resources provision, increasing software availability and reliability by eliminating impacts of hardware upgrade, maintenance, or failure.

2.1 Performance issue

The performance is always the main obstacle to prevent the wider use of hypervisor. Generally, there are three ways to improve the performance of hypervisor:

(1) Development of virtual machine technology. Virtual PC introduces dynamic code translation technology to optimize the emulation performance. VMWare virtual machine emulates PC instruction set on the PC hardware, most CPU instructions, like arithmetic instructions and memory access instructions, can be executed directly by the underlying hardware, which will gain high efficiency in virtualization. However, in order to further improve the performance of virtual machine, it is not enough to rely solely on the improvement of virtual machine itself. Support from operating system and hardware platform will optimize virtual machine performance greatly, at the expense of sacrificing transparency to some extent.

(2) Operating system support for virtual machine. Xen virtual machine developed by the University of Cambridge uses a kind of virtualization technology called the para-virtualization to improve emulation performance as well as simplify virtual machine implementation. The experimental result shows that the performance gap between xen virtual machines and physical machines is less than 3% in several common scenarios[16]. Microsoft, Novell, and Redhat are planning to support virtual machine in their upcoming release of Windows and Linux operating systems[17, 18]. The purpose of supporting virtual machine in operating system is to reduce complexity and improve performance of virtual machine.

(3) Hardware support for virtual machine. Vanderpool technology, first proposed by Intel in 2003 and later renamed to Intel Virtualization Technology (Intel VT)[19], is a hardware-level support for virtual machine provided by processor. AMD also closely followed AMD Pacífica technical specifications[20], which was published in 2005. These virtualization technologies proposed by CPU manufacturers provide new instructions and registers in the chip to assist the execution of virtual machine, which will also be an effective way to reduce the complexity of the virtual machine software implementation, and further improve the efficiency of the virtual machine.

It can be concluded that the current operating systems and hardware platforms have generally supported virtual machine. Supports from operating system and hardware can effectively improve the efficiency and stability of virtual machine, and greatly promote the application of virtual machine.

2.2 Features of Hypervisor

According to previous analysis of the hypervisor, we conclude there are four main features of hypervisor:

(1) Transparency. Transparency means software can execute in virtual machine environment directly, without being revised. Other features of hypervisor, such as hardware resource sharing, isolated environment, live migration, portability, etc, can be easily applied to any software running in virtual machine without modification.

(2) Isolation. Hypervisor enables multiple instances of virtual machine to be hosted in a physical machine. Not only software can share hardware resources via virtual machine, but also be protected and isolated by virtual machine. Each virtual machine can install its own version of software, without having to consider compatibility with the software installed in other virtual machines. Each virtual machine also maintains a separated runtime environment, while preventing software failure caused by software running in other virtual machines.

(3) Encapsulation. Whole system of a virtual machine is enclosed in a virtual hard disk, which is an ordinary file of the host machine. Through this form of encapsulation, virtual machine installation, backing up and restoring are as easy as copying and pasting a file in operating system, which can effectively reduce the difficulty of the system configuration and deployment, and increase the flexibility of software.

(4) Manageability. For virtual machine operations, such as boot, shutdown, sleep, and even add, modify, or remove virtual hardware, there all have programming interface. Via programming interface, virtual machine can be controlled by program completely. Therefore administrators have greater flexibility in controlling virtual machines, compared to manually controlling physical machines. Based on virtual machine, remote and centralized management of hardware resources can be achieved. The aforementioned live migration is another example of the manageability of virtual machine, software running in virtual machine can be

controlled to change runtime environment without being interrupted.

3. Related Work

Virtual Workspaces[8] aims to provide a customizable and controllable remote job execution environment for Gird. Traditional grid mainly focuses on job exciting, but neglects the deployment and maintenance of execution environment. Virtual Workspaces implements web service interface that conform to WSRF and GSI standard for virtual machine remote management. Client can create and deploy virtual machine on-demand via the interface.

Virtual Workspaces supports unmanned installation of legacy applications, which can effectively reduce the deployment time, significantly minimize the artificial participation in it. By allocating and enforcing resources required by jobs with different priorities, virtual machine can realize fine-grained resource provision, including CPU, memory, network bandwidth, etc. However, in some cases, applications need to run in a network environment with multiple machines, Virtual Workspaces do not suit for the situation because it cannot provide a network environment for these applications.

VIOLIN[9] is another hypervisor-based project proposed by Purdue University, which can provides the isolated network environment for applications. In VIOLIN, users can design and create virtual network topology on physical machines and physical networks. Every virtual network is isolated and independent from each other. Each virtual machine and virtual network equipment can be customized. For example, VIOLIN can build a virtual mobile IP network using Mobile IP protocol stack to test the stability and performance without actual moving of the terminals.

Similar projects like In-VIGO [21] of University of Florida and Virtuoso[10] of Northwestern University, can facilitate configuration and management of virtual machine network for network applications, and serve as the basis for emulation and evaluation infrastructure of network software. However, these projects mostly focus on runtime support, but lack of designing support which can facilitate user's creating and customizing the virtual machine network.

Intel's Internet Suspend and Resume (ISR)[22] is another interesting hypervisor systems. It can provide a personal mobile computing environment for users, without having to carry any portable device. The personal computing environment is encapsulated in the virtual machine which is stored in a distributed file system. Whenever user closes or suspends the virtual machine, ISR will copy its state back to the distributed file system so that the virtual machine can be restored from anywhere later.

The demerit of ISR is that virtual machine must be suspended before user moves, therefore virtual machine will stop running during the movement. With the use of virtual machine live migration, this restriction can be removed somehow. However, live migration requires that the source and destination machine reside in the same link, which greatly limits the mobility.

4. CIVIC Architecture

CROWN is shorted for China R&D Environment Over Wide-area Network, The major goal of our project is to provide a middleware system which enables resource sharing and problem solving in dynamic multi-institutional virtual organizations in for China e-Science community[23].

In this section, we propose the CROWN-based Infrastructure for VIrtual Computing (CIVIC), a hypervisor-based computing environment. It is a core component of CROWN Grid Middleware version 3.0. CROWN is a service-oriented grid middleware system, CIVIC can provide better support for grid service in CROWN with isolated runtime environment. The benefit of CIVIC is listed as follows: Firstly, it can offer separated and isolated computing environment for users. Secondly, it can also realize hardware and software consolidation and centralized management for computer administrators. Thirdly, it can be transparent to upper layer applications, hiding the dynamicity, distribution and heterogeneousness of underlying resources from applications.

As shown in figure 1, CIVIC can be viewed via two perspectives. From normal user perspective, CIVIC establishes a middleware infrastructure on top of raw hardware resources to provide hosting environment for virtual machine instance and virtual network instance. User can interact with the instance just like with a physical machine or physical network without having to know which specific physical machine hosts the instance. Using technologies like virtual machine live migration, CIVIC can keep virtual machine instance running from the infection of underlying hardware failure or maintenance.

From administrator perspective, CIVIC provides three modules. Firstly, CIVIC Monitor module can collect and present overall runtime status to administrator. Secondly, CIVIC Manage module can provide administrator a centralized management console for resources registered in CIVIC environment. Thirdly, CIVIC Designer module provides a visual editor which can facilitate the installation and

configuration of virtual machine instance and virtual network.

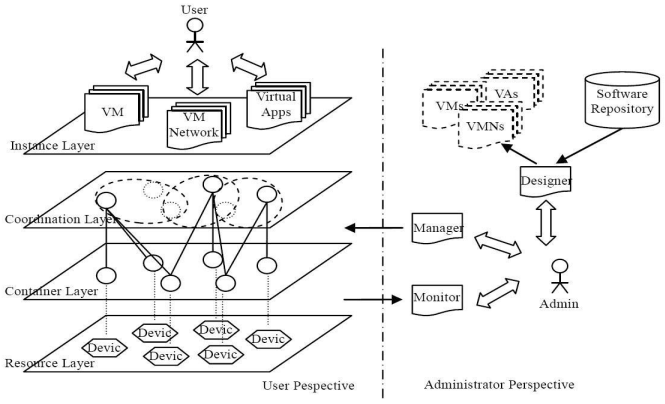

Figure 1: CIVIC Perspectives

As shown in figure 2, CIVIC can be divided into five layers: resource layer, container layer, coordination layer, instance layer, and interaction layer.

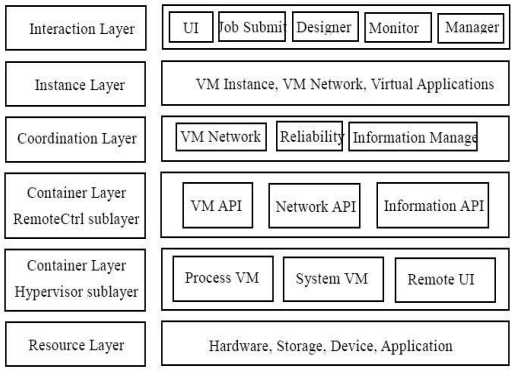

Figure 1: CIVIC Layers

4.1 Resource Layer

Resource layer is formed by multiple resource nodes. Each resource node is a physical machine. Generally, software is installed directly into these physical machines. However, resource nodes are distributed over the Internet, which may be incompatible with each other; both hardware and software may encounter unexpected failure. All these circumstances make software unreliable and vulnerable to environment changes.

In CIVIC, software will be installed into virtual machines. With the middleware infrastructure established on top of raw hardware resources, CIVIC can provide isolated and reliable hosting environment for virtual machines over dynamic and heterogeneous resource nodes on the Internet.

4.2 Container Layer

The layer on top of the resource layer is the container layer, which is composed of container nodes. Each container node is a physical machine with CIVIC Container component installed. Each container node has hypervisor software deployed that can host multiple virtual machine instances, and these instances can be controlled remotely through container interface.

Container layer can be further divided into two sub-layers:

(1) Hypervisor Sub-layer. CIVIC supports three type of hypervisor. The first one is Xen hypervisor, which can host full functioned virtual machine with operating system and application software installed in the virtual machine instance. The second one is jail virtual machine, which can provide isolated file system hierarchy for software. The third one is Java virtual machine which can host software written in Java.

(2) Remote Control Sub-layer. This sub-layer provides remote management interface, including virtual machine management interface, network configuration interface, and information query interface, etc. Each virtual machine instance can be controlled by other nodes through virtual machine management interface, including operations like booting, halting and hibernating. A tunnel connection can be established between two container nodes using network configuration interface, which can be used to connect multiple virtual machine instances hosted by different container nodes. Administrator can monitor CPU, memory usage in virtual machine instances hosted by container node through information query interface.

4.3 Coordination Layer

Container node has the ability to host virtual machine instance. However it is not efficient to deploy an instance of virtual network containing several virtual machines into a single container node. To keep reliability of the virtual machine instance, it is also important to deploy multiple replicated instances to several container nodes. Therefore, in these cases, it is essential to coordinate several related container nodes to serve one purpose.

CIVIC builds coordination layer over container layer. Coordination layer consists of multiple coordination nodes, which are special container nodes with coordinating function configured. Coordination nodes are responsible for the management of other container nodes.

As mentioned above, there are different kinds of coordination functions in CIVIC, for example, resource management for the resource registration and query, virtual network management for maintaining virtual network over a number of container nodes, etc.

4.4 Instance Layer

There are multiple nodes in instance layer, which are isolated computing environment provided for users. CIVIC support three different kinds of instance nodes: virtual machine instance, virtual machine network instance, and virtual application instance. Virtual machine instance contains a complete operating system, which is isolated by hypervisor like Xen. Virtual machine network consists of multiple virtual machines with a specific network topology. Virtual application instance contains only a single application.

4.5 Interaction Layer

This layer contains two types of interaction modules. First type of interaction module provides access interface to CIVIC instance. Users can interact with CIVIC instance nodes through a variety of interactive method, such as command line interface, graphical user interface, and Web Service interface to submit jobs to execute in virtual machine.

Second type of interactive modules is CIVIC administrative tools, including CIVIC Monitor, CIVIC Manager and CIVIC Designer. Monitor module can collect and present overall runtime status to administrator. Manager module can provide administrator centralized management for resources registered in CIVIC environment. Designer module provides a visual editor which supports creation of multiple virtual machines with a specific network topology, as shown in Figure 3.

5. Experiment

CIVIC is still in active development, we have some initial achievements. We designed four different experiments to evaluate the performance of our initial implementation of CIVIC. Our experimental environment includes two Intel P4 machine, with 3GHz CPU and 1G memory. One machine is used for the designing and creating of virtual machine and virtual network, and the other one is used to host the virtual machine or virtual network created by the first one. The experimental metric includes install time, deploy time, and boot time of different type of instance. The experimental results are shown in Table 1.

The virtual machine created in the experiment is a minimized Linux system, including the basic network tools like ftp, ssh, etc. This virtual machine occupied 136 megabytes of disk space. From the experimental results shown in Table 2, Using CIVIC to install a fresh Linux virtual machine only takes about 7 minutes, and if we use previously installed virtual machine template to accelerate the installation procedure, we can reduce the install time to 2 minutes. Taking into account that the installation of a physical machine usually takes between 30 to 60 minutes, using CIVIC can effectively speed up the system installation work and minimize manual participation in installation. Furthermore, It only takes about 2 minutes to deploy a virtual network consists of three virtual machines. In practice, it usually takes far more than 2 minutes to install and configure a physical network including several physical machines. This shows that CIVIC can facilitate installation, configuration and deployment of network-oriented applications.

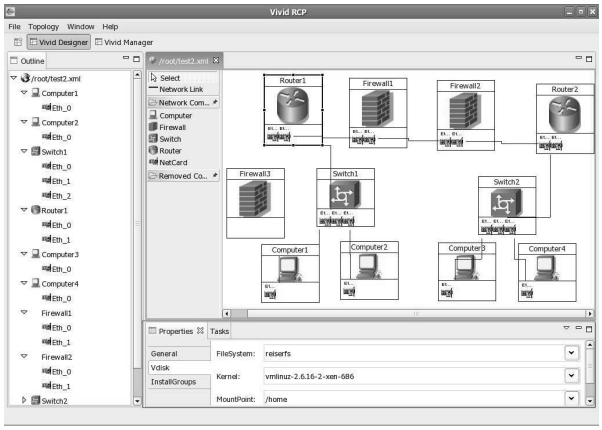

Figure 3: CIVIC Designer

Table 1: VM instance install experiments

Type of instance	Size (MB)	Install Time (sec)	Deploy Time (sec)	Boot Time (sec)
One VM instance (fresh)	136	441	45	79
One VM instance (template)	136	143	45	79
VM network with 2 VM instances	273	246	88	94
VM network with 3 VM instances	409	351	129	113

6. Conclusion and future work

This paper introduces the emergence and development of virtual machine technology, analyses typical application of virtual machine technology Then summarized up the four characteristics of virtual machine: transparent, isolation, Packaging and heterogeneity. Then through research and analysis related to the virtual machine, we presented three features of the virtual machine based virtual computing environment: being able to provide users with personal independent isolated computing environment, being able to provide centralized management functions for computer hardware and software resources, being able to offer transparent procedures for the application hiding the dynamicity, distribution and heterogeneity of underlying hardware resources. After that, we analyzed three types of typical virtual computing environment based on virtual machine system, including Virtual Workspaces, VIOLIN etc. Finally, we presented the system function design and experimental results of initial implementation of CIVIC, a virtual computing infrastructure based on CROWN.

7. Acknowledgements

This work is partially supported by the NSF of China under grant 90412011, National High Technology Research and Development Program of China (863 plan) under grant 2005AA119010, Major State Basic Research Development Program (973 Plan) under grant 2005CB321803, and National Natural Science Funds for Distinguished Young Scholar under grant 60525209.

References

[1] X. Lu, et al., "Internet-based virtual computing environment (iVCE): Concepts and architecture," in *Science in China Series F-Information Sciences 2006 Vol.49 No. 6 pp.681-701*,

[2] I. Foster, "The Physiology of the Grid – An Open Grid Service Architecture for Distributed Systems Integration," *Open Grid Service Infrastructure WG, Global Grid Forum,* 2002.

[3] D. Qiu, et al., "Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks," *Proceedings of ACM SIGCOMM,* 2004.

[4] Y. Liu, et al., "Location-Aware Topology Matching in P2P Systems," *Proceedings of IEEE INFOCOM,* 2004.

[5] Y. Liu, et al., "A Distributed Approach to Solving Overlay Mismatch Problem," *Proceedings of IEEE ICDCS,* 2004.

[6] C. Wang, et al., "Distributed Caching and Adaptive Search in Multilayer P2P Networks," *Proceedings of IEEE ICDCS,* 2004.

[7] R. P. Goldberg, "A survey of virtual machine research," in *IEEE Computer*. vol. 7, 1974, pp. 34-45,

[8] K. Keahey, et al., "Virtual Workspaces in the Grid," in *11th International Euro-Par Conference* Lisbon, Protugal, 2005,

[9] P. Ruth, et al., "Virtual Distributed Environments in a Shared Infrastructure," in *IEEE Computer*. vol. 38, 2005, pp. 39-47,

[10] "Virtuoso: Resource Management and Prediction for Distributed Computing Using Virtual Machines," http://virtuoso.cs.northwestern.edu/

[11] R. J. Adair, et al., "A Virtual Machine System for the 360/40," IBM Corp. Cambridge Scientific Center 320-2007, May 1966.

[12] "Microsoft Virtual PC 2004," http://www.microsoft.com/windows/virtualpc/default.mspx

[13] "VMware - Virtualization Software," http://www.vmware.com/

[14] M. Nelson, et al., "Fast Transparent Migration for Virtual Machines," in *USENIX Annual Technical Conference*, 2005,

[15] C. Clark, et al., "Live Migration of Virtual Machines," in *2nd ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)* Boston, MA, 2005, pp. 273-286,

[16] P. Barham, et al., "Xen and the Art of Virtualization," in *ACM Symposium on Operating Systems Principles*, 2003,

[17] "Redhat Integrated Virtualization," http://www.redhat.com/virtualization/

[18] "Windows Virtualization Technology," http://www.microsoft.com/whdc/system/platform/virtual/default.mspx

[19] "Intel Virtualization Technology," http://www.intel.com/technology/computing/vptech/

[20] "AMD Pacifica Technology," http://enterprise.amd.com/us-en/Solutions/Consolidation/virtualization.aspx

[21] S. Adaballa, "From Virtualized Resources to Virtualized Computing Grid: The In-VIGO System," in *Future-Generation Computing System*, www.krsul.org/ivan/articles/invigo_fgcs.pdf

[22] M. Kozuch, et al., "Internet Suspend / Resume," in *Workshop on Mobile Computing Systems and Applications*, 2002,

[23] J. Huai, et al., "CROWN: A service grid middleware with trust management mechanism," in *Science in China Series F-Information Sciences 2006 Vol.49 No. 6 pp.731-758*,