Advanced Science and Technology Letters
Vol.43 (Multimedia 2013), pp.52-57
http://dx.doi.org/10.14257/astl.2013.43.10

Virtual Computing Laboratories Extension with Virtual Desktop Infrastructure for Smart Campus on a Cloud Computing

Dongho Song[1], Youngpil Kim[2], Yeonjin In[2]

[1] Dept. Of Computer Engineering, Korea Aerospace Univ.,
76 Hanggongdaehang-ro, Deogyang-gu, Kgoyang-City, Gyeonggi-do 412-791, Korea
dhsong@kau.ac.kr
[2] SoftonNet Inc., 6F, Goodman Tower, 689-2,
Yeoksam-dong, Kangnam-gu, Seoul, 135-080, Korea
{ypkim, yjin}@softonnet.com

Abstract. The virtual computing laboratories (VCL) model based on Virtual Desktop Infrastructure (VDI) has been widely implemented in a small or medium scale to realize a part of smart learning systems. However, conventional physical computing laboratories with thousand of PCs are still heavily used and co-exist with the VCL. This paper proposes the extended model of VCL that could run on an integrated virtual and physical lab environment. End-users could build their own workspaces which consist of OS images, applications, and even user's personal profiles and could execute and migrate on one PC to another. The paper concludes with the discussion on the benefits, problems, and performances of the extended model of the VCL.

Keywords: computing laboratory, e-learning, virtual lab, VDI, convergence of smart devices and e-learning.

1 Introduction

This paper is about cloud computing model for extending university computer labs to outside the campus anytime anywhere. The goal was partially achieved by NCSU VCL model[1,2] but there are some unresolved issues. VDI is good for remote access for limited users who got ids of the service but it is bounded by the capacity of host server machines. VDI solutions are still cost high to apply for a whole campus and the VCL model had not showed us how to utilize campus-wide distributed PC resources.

Students yearn not only to use various PCs in campus, but to use PCs in homes or public places for seamless work from university computer labs. It can be even more effective if student's smart terminals are able to be serviced by being connected to computer labs. In the NCSU original VCL model, students can only work by connecting to the VDI on host server. If the VCL is full, they cannot continue to work in their home PCs as what they worked at their lab because the VMs cannot come out from host servers. In case of just migrating content files, such as .c, from a VM on host to local PCs, a compiler or various applications that are used at the host may not be installed on the local PCs and the profiles are not configured as the same as VM on

ISSN: 2287-1233 ASTL

the host. It seems a pity that over 95% of students have the privilege of having smart phones and accessing to the Internet, but their devices and networks are not always spatially and chronically connected with their VMs on host at VCL.

So, we need to extend the central host server based VCL model to include and utilize PCs in physical PC labs, public PCs in campus and even PCs at home in three different ways. Firstly, migrating workspace from a VM on central hosts to local PCs at campus or at home. Secondly, from VMs on a local PC to another VMs on other PC. Thirdly, connect to any VMs from end-users' smart devices. Here, the key idea is migrating user's workspaces from one VM to another VM on either running on a central host or local PCs and vice versa. By doing so, end-users could have seamless working with the same workspace on VDI and on their PC at physical lab or even at home PCs. The workspace is defined as a working image of computer desktop which consists of OS images, applications, storages, and even user's personal profiles and it can be executed, migrated, stored on one virtual machine (VM) to another.

2. Background

Burd[1,2,3], Wenhong[4], and Vouk[5] papers discussed about virtual computing lab and they compared virtual lab and physical lab. However, their basic reference model was host based VDI only and didn't include migrating a whole users' workspace from a central VDI to other VDI on local PCs in campus. Thus, the difference is that Burd model is based on host based VDI only and our model is integrated host based VDI and client PC based VDI and VMs with a whole workspace are being provisioned.

Using the virtual computing lab, students can pre-study on the laboratory work before a main class and could spend more time to discuss with instructors with what he/she had faced during the pre-study. This is an example of the Education 3.0[6] that has key notions of flipped and smart learning. Specifically in flipped learning, VoD (Video-on-Demand) systems and computer lab systems need to be integrated into LMS (lecture management system). This model has been introduced in 3 major universities in Korea as a good model of convergence of IT and education.

3. Model

The computing model on Fig.1 shows the overall system architecture and Fig.2 shows software configuration of VDIs with workspaces that comprise of in-campus and out-campus. The hardware and software layers in hosts, PCs, and smart phones perform are the same as in Fig 1 and Fig2. In other words, every hardware layers are supposed as heterogeneous for hosts and PCs. Over that the hosts or PC virtualization layers called hypervisor are installed, and on top of that Windows 7 or Linux are installed as the OS you need, and over that the Apps you need are installed. As defined earlier, a workspace is running on top of a virtual machine as a VDI.

To create a workspace, for instance, you will install Window OS and MS VisualStudio, and take a copy of a master image of everything as one. Then this master image can be used as a workspace for you and it could be executed in university lab PCs, and public PCs at libraries for you. Also, it can be performed in home PCs outside the university campus or in PCs in cafes, which is the gist of this model.

3.1 System Model

Virtual computing lab consists of the server side and the client side. On the server side, there are Z!Desktop VDI servers, Z!Sync server, Z!Stream application virtualization server, Z!BootOS server, and Z!Storage software defined storage system. These servers are all running on hosts in a rack. XEN hypervisor are running on X86 machines for VDI host server, and on top of it, a host linux, i.e. Dom0, is running and guest VMs, i.e., DomU, such as Windows7 or Linux are running. These guest VMs communicate to remote PC terminals or iPads through remote desktop protocols (RDP). On the client side, there are many PCs everywhere, and smart devices on students hands anywhere inside or outside of campus. Assuming that the client PCs are all running client hypervisor (Xenclient) locally on bare-metal hardware and linux kernel as a host and Windows or Linux as guest VMs. Note that users can connect a monitor into the PC's graphic card because the local client hypervisor can drive the graphic card directly, i.e., it is not necessary to use RDP for a local monitor. Z!Sync is a management tool to manage and synchronize images for the local client hypervisor and those stored in server. Z!Storage is a software defined storage with NAS structure. VM master images for individual students, each classes are stored centrally on Z!Storage server. It can be used for booting remote PCs at a physical lab, VMs for VDI at cloud server, VMs for local client hypervisor at home PC.

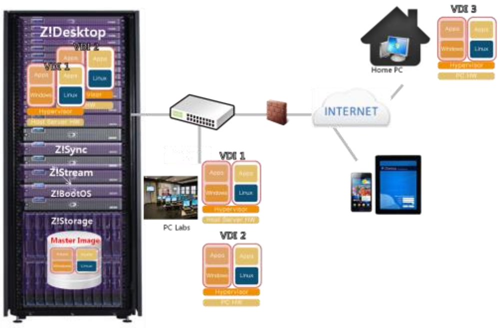

Fig. 1. A service model for workspaces.

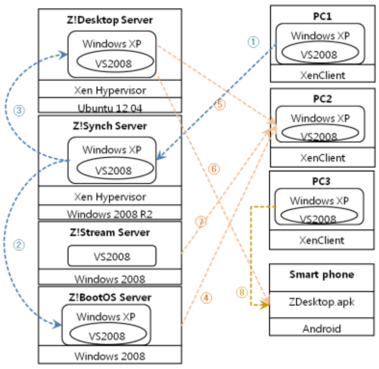

Fig. 2. An operational model.

3.2 Operation Scenarios

According to the model described above and Fig.2., a general working scenario can be divided into two phases: an administration phase and user phase. Admin phase is depicted in blue dotted arrow lines in Fig.2 and user phase is shown in yellow dotted line.

1. When the student starts a new semester, a teaching assistant creates an initial workspace for a student by preparing a clean PC1, installing required OS and applications, taking copy of the image as a workspace, and registering it to Z!Sync (line1).
2. Z!Sync store it inside Z!Storage, register it to Z!BootOS and ready for release (line 2).
3. When the student login to a VCL website with his ID on his iPhone or on his PC2 it looks for an available VM on Z!Desktop host server first. If a VM is found on Z!Desktop then it connects to it with his iPhone. Z!Sync brings the student's VM image and boots the VM in a background job (line 3). Now, RDP connection is made between his iPhone and the VM (line6).
4. If there are no available VM on Z!Desktop hosts then VCL looks for other available VM on PCs in physical lab such as PC3 and connects to it with his iPhone (line 8). The rest of the processing is the same as that on hosts mentioned above.
5. When the student enters his ID and password in remote booting the VM on PC2 using his own workspace from Z!BootOS is made by streaming the image starts the desktop as a guest OS (line 4, line 7).
6. The student will keep working on the desktop and logout. Then all the works on the desktop will be updated to Z!Sync and stored in Z!Storage server accordingly.

4. Experiments

4.1. Experiment Environment

This experiment was made on an environment of computers and networks in ordinary universities. They include 100mbps server-client's PC network, and 10Gbps backbone network. Windows 7 was the default OS, and Z!Stream, Z!BootOS, Z!Desktop, Xenclient, etc. were installed for the experiment. Note that these Z! products from Softonnet Inc. are mostly developed based on open source codes, so their strengths are on affordability. P2V is practically not feasible, so virtualization layer was installed on every PC beforehand, and student's PC image will operate on the virtualization layer. Hardware specification are as shown in Table 1.

Table 1. Hardware specifications.

	Z!Desktop Server	Z!BootOS	Z!Synch	PC1, PC2
CPU	Intel® Xeon® CPU E3-1230 V2 @3.30GHz	Intel Pentium D830 3Ghz	Intel Core 2 Duo	E6300 2GHz
RAM	16 GB	3GB	2 GB	
HDD	3.6 TB SATA	160GB SATA2	1.5 TB SATA	320 GB SATA
NIC	1Gbps NIC			

The goal of this experiment is checking whether the workspace migration is possible and meaningful. That is, the time required for the migration and the time required for remote network booting, time for storing and dispatching updated images, and these images are inter-operable on VMs on heterogeneous X86 systems.

Table 2 shows the measured elapsed time of preparing a workspace for migration in number 1,2, and 3. When the service is ready, we measured the elapsed time for bringing a workspace from Z!BootOS server to local hypervisored PC and booting the OS. It took 115 seconds until the login screen popped up. After login, it took another 47 seconds to stream the visual studio 2008 application to run. So, total elapsed time to start a seamless programming work on a hypervisored PC is 162 seconds which is quite acceptable.

Compare this elapsed time to the original NCSU VCL, Theoretically it is the same if we use the same VDI solutions and hosts. Note that NCSU VCL used VMware and we used Z!Desktop which is XEN open source based. Also, compare this elapsed time to what it would take if you build-up all your compiling work environment on a new PC from scratch. The migration of workspace is faster in order of magnitude. However, if you compare elapsed time of 40 seconds to boot WindowsXP and 3 seconds to launch Microsoft VisualStudio 2008 on a physical PC to that of 93 seconds of booting it on a virtual PC and 5 seconds to connect and another 47 seconds for streaming the application, the new model is 71% slower.

Before introducing the new model proposed in this paper, students cannot have the same workspace which consists of OS, apps, contents on their home PCs, computer lab PCs, and host servers as a uniform way making them unable to perform equally. However, after introducing the technologies and models mentioned in this paper, students could migrate a whole workspace as one single image to perform an identical work environment on different PCs at school or at home. This is a lot better than the conventional way of moving individual applications or programming contents to different PCs. Thus, the benefit is that a student's workplace can be moved around, and even if the computer system or hardware is different than what you worked on before, the work environment can be coherently connected once you boot using the same image on virtual layers. This is an unprecedented ideally seamless development environment.

Table 2. Measured data of workspace migration time.

No	Functions	Elapsed Time in seconds	Windows XP login	VS2008 launch
1	Image upload time from PC1 to Z!Sync Server	874		
2	Image upload time from Z!Sync Server to Z!BootOS Server	112		
3	Image upload time from Z!Sync Server to Z!Desktop Server	93		
4	Elapsed time for remote booting PC2 from Z!BootOS Server and VS2008 streaming		115	47
5	RDP service connection time from Z!Desktop Server to PC2		5.1	47
6	RDP service connection time from Z!Desktop Server to SmartPhone		3.1	48
7	Elapsed time of booting OS on a Baremetal PC and application launch		40	3

However, this new model exposed some issues to be resolved. Students should install virtualization layer on PCs he choose to use beforehand. A single image containing OS plus apps is usually around 20GBs, so it takes time to transfer, and requires network bandwidth to use around various PCs in and out of campus. The network bandwidth issue can be managed if more client hypervisored PCs are prepared in campus so that students can connect to them through RDP from outside of campus.

5. Conclusions

The original VCL model is quite popular to extend conventional PC labs into virtual PC labs. However, it is still relies on high cost host machines, storages , and enterprise VDIs. So, many people know that it is a good model but not affordable. Therefore, a new model is required to save TCO (Total cost of ownership) by utilizing XEN, KVM open source codes, local PC hardware resources as vehicles to run VDIs, and students' smart devices as VDI terminals. The key issues discussed in this paper are how to extend the original model into a more affordable model and would the performance of the new system is acceptable. Modeling perspective, we defined a workspace and proposed how to create, manage, execute, migrate, and store the workspace for individuals so that students can keep all their contents, apps, and even desktops to enrich their profiles. Also, the model showed up how to utilize PCs as VDI resources. We measured the performance data and it showed that the performance of the new VCL model is the same as original VCL if they both use the same VDI solutions and it is 71% slower compared to physical client PC. So, it is quite acceptable as a realistic model to apply to universities today in terms of affordability and performance. As a further study, this model will be applied to campus-wide early in the next year and more realistic performance measured data will be released in the future.

References

1. Burd, S. D., et.al,: Virtual Computing Laboratories: A Case Study with Comparisons to Physical Computing Laboratories. Journal of Information Technology Education: Innovations in Practice Volume 8, IIP-55—78 (2009)
2. Burd, S. D., et.al,: Virtual Computing Laboratories Using VMware Lab Manager Proceedings of the 44th Hawaii International Conference on System Sciences –(2011)
3. Henry E. Schaffer, et.al, NCSU's Virtual Computing Lab: A Cloud Computing Solution, IEEE computer pp. 94-97
4. Wenhong Tian, et.al,: A Framework for Implementing and Managing Platform as a Service in a Virtual Cloud Computing Lab, 2010 Second International Workshop on Education Technology and Computer Science (2010)
5. Vouk, Mladen., et al., "Powered by VCL" - Using Virtual Computing Laboratory (VCL) Technology to Power Cloud Computing, Published in the Prelim. Proceedings of the 2nd International Conference on Virtual Computing Initiative, RTP, NC, pp. 1-10 (2008)
6. Kwan, K.S.: Direction and Issues of Smart Education in Universities, Ministry of Education , Korea, (2013)