Using VCL Technology to Implement Distributed Reconfigurable Data Centers and Computational Services for Educational Institutions

Mladen Vouk, Andy Rindos*, Sam Averitt, John Bass, Michael Bugaev, Aaron Peeler, Henry Schaffer, Eric Sills, Sarah Stein, Josh Thompson, Matthew P. Valenzisi**

North Carolina State University, Raleigh, NC 27695

{vouk, sfa, john_bass, mabugaev, fapeeler, hes, edsills, sstein, jfthomps}@ncsu.edu

*{rindos}@us.ibm.com, **{matt}@ncsu.net

1. Abstract

In the context of educational institutions, small distributed data centers and labs are becoming increasingly expensive to provision, support and maintain on their own. This leads to preferences towards centralized and integrated data center resource management and network access to the resources. In turn, the data centers are undergoing a transformation driven by the nature of the equipment in them, and by the way they are used. As the computational and storage equipment becomes more powerful and more densly packed, the power and cooling needs are growing as well. One of the challenges is how to provision, manage and deliver such a large amount of physical and virtual resources in an efficient way.

In this paper we discuss how NC State University Virtual Computing Laboratory (VCL, http://vcl.ncsu.edu) technology can be used to implement distributed reconfigurable data centers and IT services in educational institutions. We use the NC State VCL implementation as a case study. It extends over four physical data centers and encompasses over 2000 computational platforms – most of them are IBM BladeCenter™ resources, some are HP, SUN and Dell platforms. VCL is an award-winning open source implementation of a secure production-level on-demand utility computing and services oriented technology for wide-area access to solutions based on real and virtualized resources, including computational, storage, network and software resources. NC State has been researching this technology and operating it in production settings since 2004. Currently, NC State VCL is serving a student and faculty population of more than 30,000. There also are VCL pilots with several UNC campuses, North Carolina Community College System, and several of out-of-state universities.

VCL technology can host practically any other environment, overlay and virtualization solution, from single-seat desktops in an operating system of choice, to high-performance computing (HPC) clusters. It is a technology well suited for hosting “cloud” solutions of almost any type. We discuss how this technology scales and what its return on investment is, and how it can deliver clouds that offer a mix of resource architectures and ensembles, including those that may integrate traditional main-frames (e.g., IBM System z and System p platforms).

2. Introduction

Small distributed data centers and labs are becoming increasingly expensive to provision, support and maintain on their own. This is especialy true in educational environments. This is leading to an increasingly centralized and network-centric management and access to these resources. The new wave is often referred to as one form of "Cloud Computing[1]." In turn, the data centers are undergoing a transformation driven by the nature of the equipment in them, and by the way they are used. As the computational and storage equipment becomes more powerful and more densly packed, the power and cooling needs are growing as well. In fact, according to some current estimates, annual costs of "care and feeding" (e.g., power and cooling, etc.) of modern equipment is approaching its acqusition costs[2]. This is strongly influencing the pilosophy of both the internal design, and the location of modern large-scale data centers. Large centers are becoming more powerful, more sophisticated, and more expensive to build. The power consumption is driving owners to locate them where power costs are lower. This, for example, may mean remote locations close to hydro-electric power sources. A side effect of this concentration of computational resources is an increased reliance on high-quality high-capacity networking infrastructure needed to connect end-users and consumers to the computational resourses located in such data centers. These centers now house thousands upon thousands of very powerful units, often blade-based, that now need even more sophisticated monitoring, configuration, and maintenance.

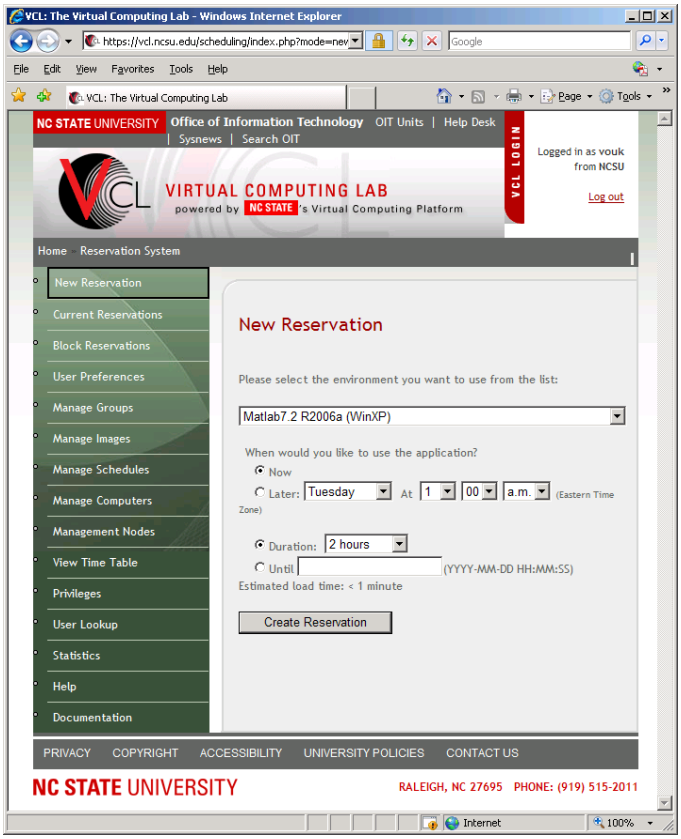

Figure 1. VCL Image/Environment reservation GUI

One of the challenges is how to provision and manage such a large number of physical and virtual resources in an efficient way. There may be an on-demand component to that, as well as a periodic one. This attention, in fact, may very well need to follow some type of a daily, weekly or seasonal cycle driven by end-user needs, by power management

[1] http://www.businessweek.com/technology/content/nov2007/tc20071116_379585.htm

[2] http://news.cnet.com/Power-could-cost-more-than-servers,-Google-warns/2100-1010_3-5988090.html

profile, or by the desire to keep expensive facility as utilized as possible. In this paper we describe

a) how the NC State University (NC State) Virtual Computing Laboratory (VCL, Figure 1) technology is being used to implement and manage a distributed, flexible and highly reconfigurable computational "cloud",

b) how this technology scales and what its return on investment is, and

c) how this technology can deliver complex homogenous or heterogenous clouds and ensembles[3], including hybrid[4] clouds, and those that may integrate traditional main-frames (e.g., System z platforms) and emerging technologies such as ensembles and Cell-based hardware..

The focus of this paper is on the technology that can help educational environments and institutions, who may have limited number of information technology support staff and limited on-site resources, offer their students state-of-the-art computational resources and applications at an affordable cost.

3. Virtual Computing Laboratory

Virtual Computing Laboratory[5] (VCL) is an award-winning[6] open source[7] implementation of a secure production-level on-demand services-oriented technology for wide-area access to solutions based on real and virtualized resources, including computational, storage, networking and software resources [1-9]. At NC State, VCL technology is being used to implement and manage a flexible and highly reconfigurable heterogenous "cloud" of computing resources that currently extends over four physical data centers and encompasses over 2000 computational platforms and over 4000 cores – most of them IBM BladeCenter™ resources, but also a variety of platforms from other manufacturers (e.g., SUN, Dell), as well as storage and applications. Some other VCL installations are based on Dell blades (Duke University) and Hewlett-Packard hardware (Western Carolina).

NC State has been researching and developing this technology since 2002, and operating it in production settings since 2004. The Virtual Computing Laboratory (VCL) concept was originally described by Vouk in 2003 [1] and by Averitt et al. in February 2004 [2]. VCL technology was developed to address the mission needs of the university and was implemented[8] by the NC State College of Engineering and Information Technology

[3] Ensembles are autonomically managed pools of like resources, i.e. ensembles require the same platform architecture for all resources within the ensemble. VCL supports multiple architectures and has a unifying mechanism, called Environment, that allows definition, construction and delivery of either a homogenous or heterogeneous cloud or a true ensemble to a user.

[4] Currently, hybrid clouds are viewed as a mix of private and private clouds. A public cloud is accessible to anyone. NC State University cloud is in part in that category, it is accessible to all NC State students, faculty and staff, and to external users who get special permission. However, VCL technology itself can support either private or public clouds.

[5] http://vcl.ncsu.edu

[6] 2007 Computerworld Honors Program Laureate Medal for technical innovation

[7] VCL is currently an Apache incubation project (http://www.apache.org)

[8] http://vcl.ncsu.edu

Division[9]. It had its first production users in the Fall 2004. Currently, NC State VCL[10] is serving a student and faculty population of more than 30,000 [e.g., 2-9]. VCL technology can host practically any overlay and virtualization "cloud", from single-seat desktops with an operating system of choice, to a hypervisor based platforms and services (e.g., VMware, Xen, or KVM), to a collection of Globus-, Hadoop- or Condor-based platforms [13 - 15] and services, to high-performance computing clusters. Virtualization of networks is also part of the VCL solution.

What is interesting about VCL (and different from a number of other solutions) is that it offers capabilities that are very flexible and diverse – they range from offering IaaS[11], PaaS and SaaS, functionalities and combinations of those, to individual and group IT services, including High-Performance Computing (HPC) services. The difference is that VCL is open source, and its 2.x and higher versions are highly modularized so that a knowledgeable end-user can replace components and is not locked into a particular IaaS, PaaS, or Saas component or solution.

3.1 Community

The NC State production VCL is distributed over three NC State campus sites and a remote site. The remote site, MCNC[12], is about 20 miles away from the NC State Campus. The sites are interconnected via redundant multi-gigabit optical links. Each site has at least one active VCL manager unit. Three production sites have the overall authentication and access management centralized. One of the sites, part of the Secure Open System Initiative (SOSI) is a test-bed site that can be "air-gapped" and is not part of the production offerings. It operates a fully stand-alone implementation of VCL that however can be, if needed, made accessible to the production cloud. It is used by researchers and developers to test and verify new ideas, and new versions of VCL. It also acts as a sand-box for "cloud" computing researchers.

Currently, the NC State VCL is serving a student and faculty population of more than 30,000 by delivering about 60,000 image (seat) reservations per semester and over 5 million high-performance computing (HPC) CPU-hours. There are VCL pilots with several UNC campuses, the North Carolina Community College System, and several of out-of-state universities – members of the IBM Virtual Computing Initiative[13].

The extent of the VCL community ranges from the production cloud that operates in the Research Triangle Park area, to a number of production pilots used by ECU[14], NCCU, UNCG, NCCCS, ODU, WCU, UNCP, NC A&T and UNCA. Plans are being formulated to make VCL available to all University of North Carolina campuses, and possibly to the

---

[9] Now Office of Information Technology (OIT)

[10] Current production version is 1.6. Version 2.0, to be released in the September 2008 time frame, will be even more flexible and reconfigurable.

[11] Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (Saas), e.g., http://www.webguild.org/2008/07/cloud-computing-basics.php, http://paastalk.com/cloud-saas-pass-market-overview/, http://www.webcloudworld.com/analysis/a-map-of-saas-paas-cloud-computing-players/

[12] http://www.mcnc.org – MCNC is not an acronym anymore, it used to stand for Micrelectronics Center of North Carolina.

[13] http://blade.org

[14] www.ecu.ncsu.edu, www.nccu.edu, www.uncg.edu, www.ncccs.ncsu.edu, www.odu.ncsu, www.wcu.edu, www.unca.edu, www.uncp.edu, www.ncat.edu,

K-12 community. Pilots also are being built in three universities in India, and there is a plan in place to implement VCL at a consortium[15] of Virginia universities (VA-VCL). There is also strong interest among several Canadian universities. VCL related research is going on at Duke, UNC-CH, Clemson and Virginia Tech.

3.2 VCL Services

The key concept and premise behind VCL is easy-to-use cost-efficient and versatile services. The base-line service is provisioning of bare-metal and virtualized resources, i.e., a suite of IaaS-type services, resources and functionalities. Extended services encompass what is sometimes called PaaS and SaaS. However, VCL is also an open source product that allows more knowledgeable users to install it and adapt it to their needs. VCL was designed to deliver over-the-network on-demand and scheduled services that enhance NC State's basic mission of teaching, research and outreach. As such, VCL is a good example of how service-oriented-architecture (SOA) can be implemented well, and how and why it can then be successful and cost-effective.

A typical user accesses VCL through an easy to use and friendly Web interface (e.g.. Figure 1) which, after appropriate authentication and authorization steps, presents the user with a set of menu options that (for a typical user) include items such as "New Reservations", "Current Reservations," "User Preferences," "Statistics, and "Help" entries. A number of additional VCL management functions are available for users with different levels of administrative privileges [8]. From a drop down menu in the "New Reservation" category (Figure 1), a user can select a particular environment of interest (e.g., a MATLAB single-server image) and request it for a given period of time either immediately, or at some other future time. The image or image environment is loaded when requested for the duration requested on either an implicit automatically assigned resources (e.g., for general IT services) or on an explicit set of resources (e.g., a tightly couple cluster of specific nodes for an HPC experiment).

VCL also has a network-oriented (service) application programming interface (API) that allows remote applications, middleware and operating systems to access the same functionalities through a publishable service. This allows automatic augmentation of end-user needs and seamless addition of resources provided the end-user platform is configured to do so. This is very much in the spirit of "cloud" computing where, ideally, end-user would not be aware that external resources may have been added to his/her personal platform in order to complete the requested task.

From the end-user perspective, VCL offers a series of services range from single-seat desktop type offerings with access to either routine or specialized computational resources and applications, to groups of seats that can be reserved for a particular time-slot, to reservation of one or more servers, to reservation of homogenous or heterogeneous aggregates (or " bundles") of computational and storage resources called "environments" which are the building blocks of "virtual clouds" that VCL supports, to long-term reservation of research clusters, to high-performance computational (HPC) cluster and facilities, and so on. On a typical day during a semester, about half of our

[15] http://gmuproject.pbwiki.com/VAVCL

resources are providing HPC service. In this context, a "seat" can represent either access to a sole-use virtual or bare-metal resource (e.g., a Windows or Linux server) that in turn can be used on its own, or as a management node for a group of services or resource, or it can represent access/portal to an already pre-configured shared service or resource (e.g., access to an account on an IBM System z machine, or access to HPC clusters via a sole-use or shared login-node).

One of the primary characteristics of VCL is that it can dynamically change its configuration and move resources from one type of service to another. In the current implementation, we distinguish two major groups of service categories: HPC computing services and other services. The former consist of access to cluster-based resources controlled through an HPC scheduler such as LSF[16], access to shared memory resources, and in special cases to supercomputers.

Other services include:

a) access to single-node bare-metal or virtual computers - typically in the "desktop" mode but could be in some other way - these resources typically come to an end-user with administrative privileges (root access) this is further discussed in the security section of the paper,

b) ability to make reservation for a group of desktops for a particular time-slot (e.g., for use in a class during a class period, or a bank or office during working hours),

c) ability to reserve one or more (bare-metal or virtual) servers,

d) ability to reserve aggregations of tightly-coupled and/or loosely coupled computers and servers either of the same type (e.g. blades for a computational test cluster) or an integrated set of diverse components (e.g., a hybrid aggregate might consist of a web-server, a data-base server, a System z resource running applications, and a cell-cluster to perform some related analytics), and

e) ability to use the VCL to request a portal into HPC cluster-resources and submit batch or real-time jobs to such an environment.

Access to the reserved resources can occur in any way that suites the end-user. For example, Windows desktops are typically accessed through an RDP client, Linux single-seat resources may be accessed through X-windows and/or ssh, a web-service may be accessed through a web-browser or through another application, while HPC resources can be accessed either through a personal login-node reserved via the VCL drop-down menu, or through a set of communal login-nodes, or through a web-interface.

It is worth noting that from the perspective of educational institutions of higher learning, even research intensive universities, there is a huge advantage in having a collection of resources that can morph from providing relatively simple IT services, to providing research clusters and HPC services. It increases utilization of the resources, and provides plasticity that invariably results in reduced cost of both operation and ownership.

Functions that VCL offers to someone with management privileges range from image and environment creation and modification, to mapping of bare-metal and virtual images to underlying hardware resources, to grouping of hardware resources, to monitoring and

[16] http://platform.com/Products/platform-lsf

tracking of the VCL usage, capacity and performance, to an ability to manage privileges of different users and groups [8].

3.3 Architecture

VCL is a 50+ kLOC product developed at NC State University. It is currently in the incubation stage with Apache. In addition the VCL code, a VCL implementation leverages a number of additional open source and commercial products. They will be discussed as we discuss VCL components.

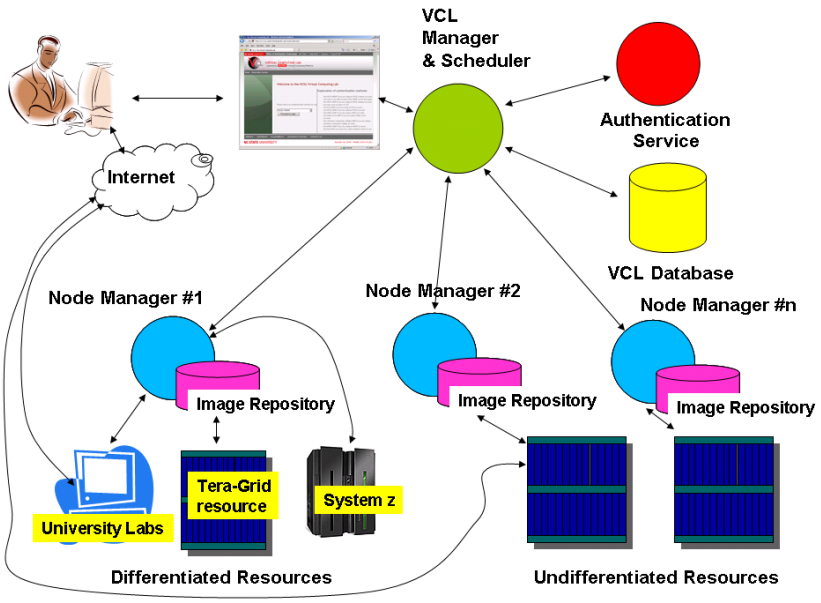

Figure 2. VCL Architecture

VCL architecture is all about components. The top level architecture of VCL is illustrated in Figure 2. There are seven major groups of components (or subsystems) in the VCL architecture:

a) an end-user access interface (web-based and API based) subsystem,
b) an authentication service,
c) a VCL Manager which manages user requests and includes a resource scheduler, authorization, security, multi-site coordination, performance monitoring, virtual network management, etc.,
d) VCL database,
e) a node manager which manages local installation resources and loads VCL images,
f) an image repository with images, and
g) computational, storage and networking resources.

VCL architecture abstracts resources at several levels: at the application and operating system level via images and hypervisors, at the hardware location level (via VCL manager and nodes), and at the network level (via virtual networks, VLANs, VPNs, etc.). VCL technology is in many ways an active and integrating "shim" between more traditional virtual environments (e.g., operating systems, and networks, including most IaaS, SaaS and PaaS solutions) and the underlying resources – typically a composite of the software and hardware stack. In keeping with the SOA philosophy, there are options

to either replace a component with a functionally equivalent one, or use alternatives within the same installation.

3.3.1 Images

The basic IT services oriented delivery mechanism of VCL are its "**images**". Conceptually, an image is a software stack that incorporates

a) any base-line operating system, and if virtualization is needed for scalability, a hypervisor layer,

b) any desired middleware or application that runs on that operating system or on a hypervisor, and

c) any end-user access solution that is appropriate.

Images can be loaded onto "bare-metal", or onto an operating system/application virtual environment (hypervisor) of choice. When a user has the right to create an image, that user usually starts with a "NoApp" or base-line image (e.g., Win XP, Vista or Linux) and extends it with his/her applications.

A special case are images that have one or more sub-images, i.e., images that are always loaded along with the parent or master image. We call these composites "environments." When a user constructs such composite images (aggregates of two or more images), the user extends service capabilities of VCL. In fact, "environment" capability of VCL may be used to construct specialized services and virtual clouds of almost any type. A user can have either sole use of one or more hardware units, if that is desired, or the user can share the resources with other users.

VCL supports multiple **image formats** and has the potential to use many more. The most common types are kickstart based installs for Linux distributions,, disk images using partimage[17], VMware disk images for both VMware Free server and the ESX standard. In addition to these image types, VCL also brokers remote access to standalone machines by interacting with a *vclclient* daemon installed on the remote device, or another appropriate service. This is ideal for increasing utilization of traditional academic computing lab machines while they are idle, as is done at NC State, or for including resources that otherwise are not available on campus, e.g., access to IBM System z resources.

In a classical lab setting a student may not have administrative control over a machine. In full VCL setting, users typically do have administrative privileges. However, after each use, a VCL machine is re-loaded with a clean image. Similarly,–non-administrative mode – is a good way of approaching integration of resources such as IBM System z LPARS and System p virtual machines, or other resources where one may only have access as a non-privileged user.

A very important characteristic of VCL images is that they have a lot of **meta-data** associated with them. An image has an identifier, location, name, owner, memory footprint, speed of access information, licensing information, hardware requirements, loading time, access permissions, and so on. Images can be associated into image groups. Images can be mapped onto particular resource ("hardware") groups or even individual

[17] Partimage: http://www.partimage.org

computers. Association of the meta-data with an image is made primarily via the VCL data-base. Image formats depend on hardware/hypervisor on which they run [8]. Use of composite images to construct "virtual clouds" and cloud services is discussed later in this paper.

In addition, VCL collects a lot of information about the image usage and availability. That allows managers to chose appropriate image pre-loading schedules and profiles, recognize and track down problems, and increase the security of the system. VCL users can access statistics for non-HPC images via the "Statistics" menu bar. Currently, VCL image pool is about 600 large, but only about 120 or so of the images are used frequently and another 70 to 80 less frequently. During Spring 2008 VCL end-to-end service reliability was about 0.99, and most of the reservations were of the "now" type with an average duration of a reservation of somewhat less than one and a half hours.

3.3.2 Resources

We distinguish between two types of resources: undifferentiated and differentiated. **Undifferentiated** resources are those that can be reconfigured and reloaded at will with whatever suites the end-user. Differentiated resources are pre-configured, but can be made available to the end-user at will or on schedule. For example, a group of blades that can be loaded from scratch ("bare metal") with Linux, Windows or some other operating system and applications on short-notice represent undifferentiated resources. Similarly, a group of servers that is already loaded with a hypervisor (e.g., VMware ESX) and can receive any virtual image of choice represents an undifferentiated resource for that that type of virtual images. When a user is finished using the resource it is again returned to the pool of undifferentiated resources.

On the other hand, a group of machines that may be located in a university computing laboratory and that are made available to users over the network when the laboratory is closed, may be classified as **differentiated** if the end-user does not have the right to re-load them at will, i.e., can use them only in the already configured (differentiated) state. Similarly, access to an LPAR or an account on an IBM System z resource may be considered as access to an already differentiated resource. In this case, when the user is finished with the resource(s) they are also returned to the pool but remain differentiated. Of course, any web-services offered through VCL affiliated resources fall into the same category.

VCL offers its administrators the ability to incorporate with ease new hardware, to administer meta-data about the hardware, and the hardware itself (e.g., reload, default-image, etc.). Hardware can be grouped into logical groups that may reflect its properties (such as inter-connectivity, processor type, ownership, etc.), and of course images can be mapped onto any hardware component, some hardware groups (e.g., those that are tightly coupled through a high-speed-interconnect such as Myrinet), and assigned to specific user groups.

Of special interest is the already mentioned ability of VCL to work with differentiated resources. At NC State, such resources often take the shape of regular computing laboratory machines that are made available during the time the labs are closed. Access and preparation of these resources is effected through the *vclclient* agent. However, exactly the same principle can be used using service channels that may be offered by

other differentiated resources. These resources may take the shape of platforms that allow direct or indirect login (perhaps through a platform-specific method) and appear to the end-user as desktop augmentations, or they may take the form of seamless augmentation of the computational requests from the end-user, perhaps through a background batch job submission process. In this way, and so long as this service API is well defined, any type of hardware can be incorporated into a VCL cloud. Depending on what the interface allows, VCL may be able to reserve and manage the resources, or perhaps only use them.

One such example of special interest is incorporation of IBM "Blue Cloud" ensembles[18] – intelligent aggregates of self-managed hardware, and incorporation of classical HPC resources such as IBM System p computer, business "mainframe" resources such as IBM System z resources, and a variety of specialized hardware solutions from other manufacturers such as Dell, SUN and Hewlett-Packard.

In general, thanks to its modular nature and service-orientation, it is easy to integrate VCL with any other "cloud" solutions either in a peer-to-peer mode, or in a hierarchical mode.

3.3.3 Resource Management

VCL resource management is performed using a combination of locally developed code and off-the-shelf products. The latter include IBM xCAT[19] and IBM Tivoli Monitoring[20], an open source web server (Apache), and an open source data base (MySQL). All components can be (and are) distributed (Figure 2).

There are two principal parts to the VCL image manager: our open source code for resource discovery, scheduling and mapping, and an image loader and platform manager (in the current implementation this are xCAT, Cobbler[21] and VMware[22] loaders). VCL node manager needs to handle both bare-metal images and virtual images (i.e., images that can load on top of a hypervisor). This is done by providing an interface (API) to the appropriate lower-level middleware. Currently, VCL interfaces either to xCAT or Cobbler for bare-metal loading, and the VMware loader for loading of guest operating systems and applications onto free VMware and ESX platforms. However, the architecture allows substitution of different (possibly proprietary) components. We have a prototype running Cobbler as a Linux loader, and we are in the process of integrating Xen and KVM image managers. In the context of VCL the hypervisor base-line is just another bare-metal load. For example, an administrative manager could request loading of a ESX bare-metal server image, and then individual users can make requests for a particular virtual image that would be loaded onto that server. This process is automated, i.e., when additional ESX servers are needed, VCL can request them automatically to keep up with the capacity needs. An alternative is to simply consider the underlying "hardware" as already having the hypervisor layer (pre-loaded and fixed) and just use VCL to manage deployment of virtual images onto that group of resources.

---

[18] https://spaces.internet2.edu/download/attachments/8817/ComputingAsAService08.pdf?version=1

[19] http://www.xcat.org/

[20] http://www-306.ibm.com/software/tivoli/products/monitor/

[21] http://cobbler.et.redhat.com/

[22] http://www.vmware.com/

We use both approaches in practice. We have banks of undifferentiated hardware onto which anyone with appropriate permissions can load a VCL managed bare-metal image, and then use that image as they wish. If the image is an ESX server, the user can manage any additional guest operating system loads onto that image. Alternative is to associate that server with the VCL resource pool as a differentiated "resource" that is now managed by the VCL scheduler. That resource can now host any appropriate number of guest operating systems of choice. We also have banks of blades already pre-loaded with GSX or ESX images which are integrated into the VCL "hardware" pool. One thing to remember is that the bare-machine loads are a necessary functionality since bootstrapping of a hypervisor-based pool still requires loading of hypervisors onto the hardware of choice.

The level of automation available to VCL users depends on the user category. We distinguish four categories of users: Service end-users (e.g., students in a class), service integrators and extended image creators (e.g., faculty, teaching assistants), basic-services developers and base-line image creators (e.g., specially trained VCL staff), and VCL developers. The highest level of automation (with the least number of manual intervention options) is available for the first group of users. The ability for manually control cloud functionalities, services and resources increases as one progresses to more enabled user category levels.

For the first category of users, practically everything is automated – the user just needs to select the image (application, operating system, service, etc.) they wish and that image is automatically loaded on an appropriate platform and run. Even the selection process can be automated using the API. If the image is created for bare-metal loading (usually for heavy duty applications with large memory and CPU footprint), hardware is automatically selected from the available pool. If the image has a lighter footprint, it is often loaded on top of a hypervisor.

The second category of users is assumed to have a higher level of IT knowedge and VCL training. Such users typically construct images and environments (two or more clusters of images) for asynchronous and synchronous use by others, e.g., individual students or a class, manage their access permissions, and similar. If block reservations are used (e.g., need 20 "seats" of image x all Mondays of the semester from 10-11am), the reservations can be pre-configured by service integration users and automatically scheduled for the needed periods. Alternatively, manual loading on the part of an authorized user is possible (but not usually recommended). Also, construction of new extended images is a manual process – e.g., XP is the base-image, but to that application Y and Z are added, image is saved for use by Q students at a time in course D. A lot of information is accompanies the images. This includes licensing information, access permissions and so on. At the time of the image creation, a decision needs to be made whether it is a bare-image load or a virtual image load (and on which hypervisor). At image deployment time, and depending on image parameters (e.g., memory footprint, CPU speed, etc.) the image is automatically loaded in either bare-metal form or onto a hypervisor that operates on hardware that has enough memory, CPU power, and coupling to its sibling images (if any), and that has enough capacity left to run the image. Capacity information about all hardware is automatically maintained by the VCL scheduler in its database. Category 2 users also can explicitly specify pools of hardware units on which they wish to run if this is important, otherwise that choice is left to the VCL scheduler.

3.3.4 Node Manager and Network Set-up

One of the key features of the undifferentiated VCL resources is their networking set-up. It allows for secure dynamic reconfiguration, loading of images, and for isolation of individual images and groups of images. Every undifferentiated resource is required to have at least two networking interfaces. One on a private network, and the other one on either public or private network depending on the mode in which the resource operates. Also, for full functionality, undifferentiated resources need to have a way of managing the hardware state through an external channel –for example through the BladeCenter™ chassis Management Module.

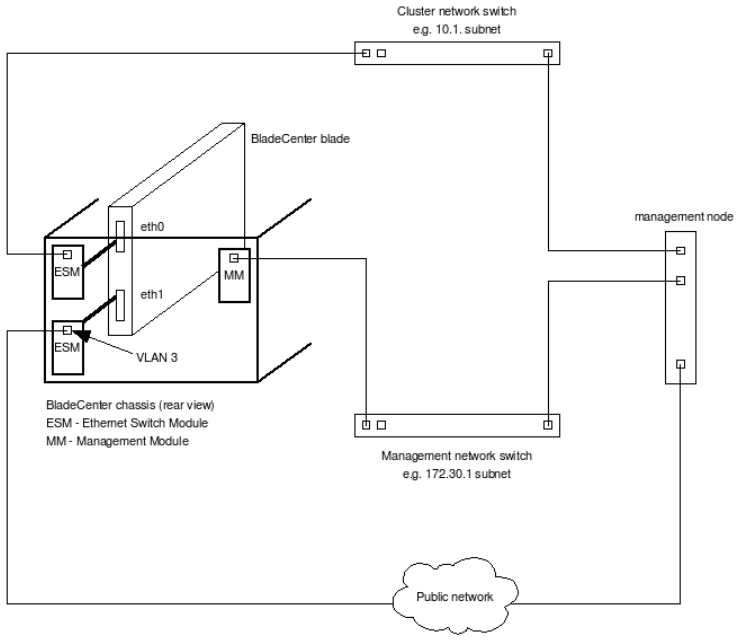

Figure 3. One of the dual-use physical network configurations.

Figures 3 and 4 illustrate two basic VCL network configurations. The first configuration (Figure 3) is for environments where seats/services are assigned individually or in synchronized groups, or when we want to assign/construct an end-user image aggregate or environment where every node in the environment can be accessed from a public network (e.g., an image of a web server, plus an image of a data-base, plus an image of an application, or a cluster of relatively independent nodes). Typically eth0 interface of a blade is connected to a private network (10.1 subnet in the example) which is used to load images. Out-of-band management of the blades (e.g., power recycling) is effected through the management network (172.30.1 in the example) connected to the MM interface. The public network is typically connected to eth1 interface. The VCL node manager (which could be one of the blades in the cluster, or an external computer) at the VCL site has access to all three links, that is it needs to have three network interfaces. If it is a stand-alone server, this means three NICs. If management node is a blade, the third interface is virtual and possibly on a separate VLAN.

It is worth noting that the external (public) interface is on a VLAN to provide isolation (e.g., VLAN 3 for the public interface in Figure 3). This isolation can take several levels. One is just to separate resources, another one is to individually isolate each end-user

within the group by giving each individual resource or group of resources a separate VLAN – and in fact end-to-end isolation through addition of VPN channels. This isolation can be effected for both real and virtual hardware resources, but the isolation of physical hardware may require extra external switching and routing equipment. In highly secure installations it is also recommended that both the private network (eth0) and the MM link be on separate VLANs. Currently, one node manager can effectively manage about 100 blades operating in the non-HPC mode.

The second configuration (Figure 4) is used when the blades are assigned to a tightly coupled HPC cluster environment, or to a large overlay (virtual) "cloud" that has relatively centralized public access and computational management. In this case the node manager is still connected to all three networks – public, management and image-loading and preparation private network, but now eth1 is connected (through VLAN manipulation, VLAN 5 in Figure 4) to what has now become an Message Passing Interface (MPI) network switch. This network now carries intra-cluster communications needed to effect tightly couple computing tasks usually given to the HPC cloud. Switching between non-HPC mode and HPC mode takes place electronically, through VLAN manipulation and table entries; the actual physical set-up does not change. We use two different VLANs to eth1 to separate Public Network (external) access to individual blades when those are in the Individual-Seat mode (VLAN 3 in Fig 3), from the MPI communications network to the same blade when it is in the HPC mode (VLAN 5 in Figure 4).

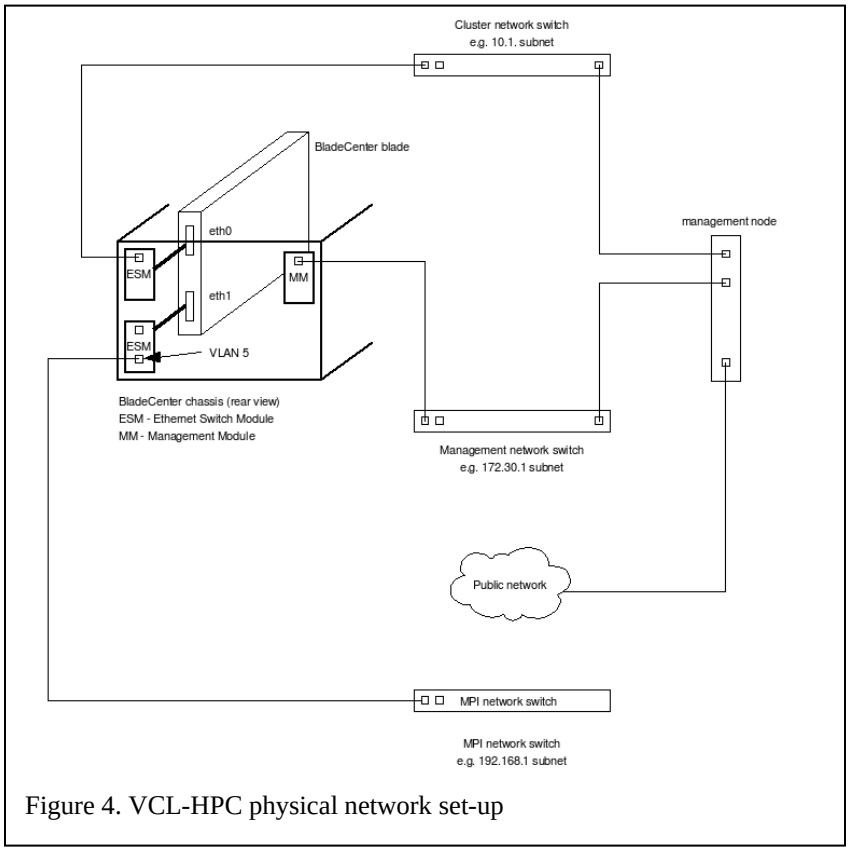

Figure 4. VCL-HPC physical network set-up

For both configurations it's recommended to have separate networks for:

a) Cluster/private use, where image loading and other private operations happen (such as NFS mounts), and
b) Management use, where management node communicates with Management Modules of a chassis (e.g., via xCAT commands)

Virtual cloud components and services based on the first configuration can be easily distributed across different data centers – image loading is effected by different node managers, but access and scheduling is done through one scheduler and interface. Tightly coupled HPC configurations are usually limited to a single data center, and also may be limited to specific equipment groups, because they often require special high-performance interconnects (such as Myrinet[23]). Loosely coupled HPC-based "clouds" can be distributed over multiple physical data centers, but may require VLAN and IP address tunneling.

Scaling, security and effective management of VCL to datacenters containing thousands of blades requires not only appropriate cooling and power, but also an appropriate networking infrastructure. A solution for scaling the VCL network infrastructure should consider explicit conservation of network resources such as VLAN IDs and increased network stability via the reduction in size of Spanning Trees required for Data Link Layer topology maintenance. The switches can be preconfigured such that the VCL provisioning software can access needed resources in a deterministic fashion. Part of security can be provided via stateless Access Control Lists that prevent individual physical resources and Virtual Machines (VM) from exchanging data packets with each other or to allow certain resources to communicate with a subset of other resources to allow for collaboration between resources when needed. To ensure system availability a redundant network architecture is recommended. For example, the VCL backbone layer could consist of two enterprise/carrier class layer 3 switches tethered to n + 1 pods of distribution switches each comprised of enterprise/carrier class layer 3 switches.

3.3.5 HPC Services

The NC State VCL implementation extends from "single seat" desktops to HPC offerings. In the case of distributed memory HPC, blades are loaded with the HPC Linux computational images with Gigabit Ethernet interconnect. This group of services operates on clusters of machines/blades isolated through an HPC VLAN. A subset of nodes have additional Myrinet interconnect. The HPC Service Environments include management, computational resources, and login nodes. The only public access is through one or more login nodes. Through VCL, HPC users can request a personal HPC login node image which they can then use to track their jobs in real-time. At NC State the scheduler of choice is LSF, but any other job scheduling solution can be used.

While about 1600 processors are in the HPC mode on an almost permanent basis, there are another 400 to 500 processors on campus, and another 2000 off campus that can be moved in and out of the HPC configuration. During the semester breaks and in the summer when there is less need for non-HPC resources, a number of VCL non-HPC resources are moved into the HPC mode. This significantly increases the reuse of blades and allows the infrastructure to be shared, leading to greatly enhanced usage levels and economy.

[23] http://www.myri.com/

3.3.6 Storage

Of course, there is also a considerable amount of storage associated with the VCL operation and architecture. VCL users can

a) access storage on platforms on which images run (typically blades),
b) corporate network-based storage (backed up), and
c) storage on their own access platforms (e.g., laptops).

In addition, HPC images have access to about 3 TB of shared (scratch) memory storage, and another 32 TB of mass storage (backed up). NC State VCL images are constructed with storage access modules appropriate for the environment and institution they operate in. At NC State University, we equip Windows images with applications that access our corporate/enterprise level storage, as well as any network accessible storage anywhere. WolfCall[24] is an application that lets our users access their NC State allotted corporate storage space (AFS-based). Our Linux/SunOS images have a similar access to our AFS space. We also construct our images with more general GUI-based applications, such as SSH Secure File Transfer Clients or PUTTY[25]. Images that use RDU have an added option of using RDU-based remote storage on the access platform storage devices (e.g., end-user drives, memory key, and similar).

3.3.7 Security

As with any networked system, security is of great importance and continuously needs pro-active and preventive action. The VCL uses simple but effective security measures in both authentication and authorization of services, as well as in implementation of end-to-end security. In the four years in production operation, VCL installations did not experience a single major security issue. In part this is due to the security configuration which we use, but in part it is also due to the operational profile of our users.

Authentication is the first step into VCL. At NC State it is via a solution called Wrap[26], but typically it is via affiliation-based (institution-specific) LDAP or UNC Federation Shibboleth. The latter two methods are used by other UNC participants. In addition VCL support local web-server based authentication. The environment-level access permissions are defined at image creation time and end-users see only those images they are authorized to use.

After a user is authenticated and allowed to make a VCL reservation, VCL **IP-locks**[27] the provisioned environment to the end-user IP address using OS level firewall. In a highly-secure variant, additional VLAN isolation would also be implemented. During the reservation process, and in the normal reservation mode, the user is required to acknowledge the reservation request. During this acknowledgment the web application captures the visitor's address. The management node then proceeds to modify the OS level firewall(s) (and in the highly-secure environment, manipulate VLANs) of the assigned reesources by opening the correct service port(s) for that remote IP address to

[24] http://www.eos.ncsu.edu/wolfcall/

[25] http://www.chiark.greenend.org.uk/~sgtatham/putty/

[26] https://webauth.ncsu.edu/wrap/

[27] Internet Protocol (IP)

connect to the VCL compute node. In this way we can control the nature of the access to the VCL cloud – from private to partially private, to totally public.

A time-out paradigm is used to handle reservations that have been made but not consumed, as well as any idle user-to-VCL links.

Currently the primary direct access methods are ssh (secure shell) to Linux-based systems, and remote desktop protocol (RDP) to Windows-based systems. Both approaches incorporate a level of **encryption** and therefore add to the end-to-end security. Other access methods are possible.

In the case of Windows machines, we also implement **one-time passwords** as an added security method. The Linux environments typical make use of existing enterprise-level authentication infrastructure, or can also use standalone account mechanisms.

Another important security factor is that the undifferentiated VCL compute blade resources operating are **re-imaged** once the end-user reservation is over. This removes any residual data files and possibly malicious add-ons from one user to the next.

Still there are other security methods available, such as using VPN's end-to-end, private VLAN's, ssh-tunneling, etc. For example, NC State has in place VPN facilities that are open to all students, faculty and staff. This is used to add to the end-to-end security between the remote station and the inside-the-campus VPN end-point. This point can be moved into the data-center VCL domain.

As discussed in the previous section, isolation of individual reservations from each other is an option. It is important that this solution be an automatic part of the image-associated security. For example, VCL allows a user with image creation rights to view and modify image attributes. An image owner can set the groups that have the right to access the image and can also set licensing limits for the image. There is a separate menu (under Manage Groups) that allows mapping of the images onto specific hardware groups. That provides an additional option regarding security.

We also monitor all VCL networking traffic. If an issue is reported, such as excessive incoming or outgoing traffic or attack attempts, we have the option of blocking the affected VCL reservation IP addresses. In addition, since we have the full record and history of who has made reservations and from where, and who has created or modified images, we have considerable ability to track down potential culprits (intentional or inadvertent). Since the Fall 2004 when VCL went into production we have not had a single report of intrusion or malicious misuse of the facilities.

Recently NC State has been awarded federal funding to address security concerns in Open Systems and open-source software, thus the Secure Open Systems Initiative (SOSI)[28] has been established with one of the goals being a security hardened version of VCL. Within that context we are working towards the following, more comprehensive security vision of VCL:

- End-to-end isolation (on the public side, i.e., an end-to-end isolation of communications to and from individual VCL (reservation) environments (e.g., via VPN into the a Data Center, and then via VLANs to the images, whether virtual or bare-metal,

[28] http://www.engr.ncsu.edu/news/news_articles/sosi.html

- Back-path (private network) isolation and blocking – back-paths are also VLAN-ed and during a reservation closed off (from the manager side)
- Encryption of data on VCL images would be used routinely for data on VCL units.
- Watermarking and security certification of both VCL images, including hypervisors.
- Post-reservation scrubbing of storage (real or virtual) after high-security unloads
- Additional application, operating system and network-based security.

3.3.8 Performance, Reliability and Fault-Tolerance

VCL presents an easy-to use web-based interface. Our users find it appropriate and are satisfied with it. At initialization, VCL scheduler typically pre-loads all available resources with platforms and applications using the resource's "standard" load profile based on VCL's long-term operational profile. After that, scheduling switches to most-recently used mode. This adapts to short-term changes in the operational profile of the VCL. We are working on some more advanced and pro-active scheduling alogorithms.

When a user makes an on-demand reservation two things can happen: a) the requested image or service is already pre-loaded, which case it is made available to the end-user within 30 to 60 seconds, and b) the requested image needs to be fetched from the appropriate image library, prepared and loaded, in which case it is made available in the 6 minute to 20 minute time-frame depending on its size and whether it is a VMware image or a bare-machine load image. When users make advanced reservations requesting a particular image, that image is pre-loaded to so that it is available when the request period starts.

During the Fall 2008 (20th August to 31st December 2008) there were 82,298 non-HPC VCL reservations, 78,944 (or about 96%) were on-deman ("now") requests. A total of 2,907 requests were not satisfied for a variety of reasons ranging from unavailability of resources (oversubscription), to failures and faults that basically resulted in the reserved computer failing to get prepared for user (1,347 cases). In the former case, the user is offered to schedule a later reservation, in the latter case the system attempts several loads on possibly different resources. Failures are logged and if the fault is in the hardware instead of the image, the hardware is taken off-line for later inspection. A lower bound on the 2008 VCL service reliability (including HPC services) was in the 96-99% range.

If a run-time failure occurs in the hardware, rescheduling of the image is left to the end-user. However, if an image is a virtual one, then depending on the hypervisor an automatic fail-over may be possible. During the Fall 2008 67,812 (82%) reservations had image load times of less than 2 minutes, and 14,486 (18%) load times in excess of that. The most popular non-HPC software in 2008 was Maple.

3.3.9 Licensing

VCL is an open source product. It is currently being distributed under the Eclipse and Apache licenses. VCL is very conscious and protective of intellectual property and licensing rights. Images created under VCL must have licensing resolved at the time of image creation. Images can use network-based license servers, such as FelexLM or KeyServer, or can limit the number of images that are active according to the number of licenses purchased for the image.

4. Utilization, Scaling, and Economics

VCL was developed by NC State as a response to our realization in the 2002 to 2004 time-frame that the then existing information technology solutions were not keeping up with mounting needs of our users and the primary mission of our university: excellence in education, research and outreach. Our students, faculty and staff needed very flexible and quickly deployable access to the latest computer-based teaching and productivity tools, advanced and often industrial strength applications, and extensive research computing facilities. Yet, those services needed to be delivered in a cost-effective way, and with the flexibility that would allow the solution and the services to change with times and grow without being vendor- or approach-locked and without requiring increased support staff requirements. We realized that a paradigm shift was needed and that the three key ingredients for achieving this were simplicity, flexibility and leveraging of the commodity hardware and software offerings.

The services that we needed were more advanced than what could be, or was practical or cost effective to, run on personal workstations or laptops (e.g., because of licensing, cost, or hardware issues), but at the same time did not reach into the rarified atmosphere of the supercomputers. However, this middle layer of needs may, on occasion, extend further into the personal access device space (e.g., when thin clients are used), or may reach more toward the supercomputing resources when complex scientific workflows are in question. Therefore, our solution had to be not only layered and scalable, but also component-based [e.g., 10] and thus extensible. The underlying hardware was an issue. In the quantities we were looking at, rack mounted (xRU) equipment did not scale well and its maintenance and management was too labor intensive.

We did an extensive study of the commercial computational solutions available at the time, and we came to the conclusion that the only compact, scalable, extensible and reconfigurable solution that fit our needs needed to follow the blade-based model. The only product that at that time was functionally appropriate, had the right form factor and reconfiguration characteristics, and had the appropriate performance to cost ratio (including power consumption), was the then newly announced IBM BladeCenter™. Since then, blade-based solutions have become recognized as the “standard”. However, what was not available at that time was the scheduling and management software that would utilize blade-based data center equipment to manage and deliver highly reconfigurable and flexible computational and application services. This is what prompted development of the VCL solution. We believe that in the context of educational institution needs this solution is still ahead of both open source and commercial solutions available today.

One of the key questions in all this was whether VCL-based solution is cost effective. Over the last five years, we have found out that not only it is cost-effective, but that in it essence it is what "cloud computing" is. Hence by answering the question of whether VCL is cost-effective, we can also explore the question of whether "cloud computing" is cost effective. VCL addresses the cost issue in a number of ways:

a) VCL considerably increases **utilization** of resources by allowing dynamic re-allocation of excess undifferentiated resources to the purpose that is in need of additional capacity (Figure 2). As already discussed dynamic sharing of the resources among laboratories, HPC users and other users keeps the resource utilization very high (in the 70-80% range). In addition to that, augmentation of laboratory desktop computing resources with high-performance VCL services extends the life-time of laboratory machines (by as much as one third) since they do not have to be replaced as frequently. Finally, gradual migration of HPC-grade blades to less demanding use over an 18 to 24 month period, and extended use of these blades (up to 5 years) not only increases capacity, but also increases return on investment.

b) **Lab space**. Another thing to note is that a typical NC State bare-metal blade serves about 25 students seats – 25:1 ratio. With hypervisor the multiplier can be an additional factor of 2 or more, possibly reaching into dozens depending on the virtualization method, application footprint, and hardware charateristics. This is a considerable improvement compared to the 5:1 or 10:1 ratios used for classical computer labs. However, this does not mean that classical computer labs may disappear. They have a considerable social networking, collaboration and community learning function. However, new facilities may not need to be built even as the student population grows since many of their more routine functions would be picked up by VCL.

c) VCL allows **power saving** management based on the operational profile of the resources (see the discussion in the next subsection).

d) VCL considerably **reduces the routine management effort** for the undifferentiated resources through real-time monitoring of the resources, remote management of its states, and through implementation of fail-over policies that make use of the redundancy in the resources. The routine management, maintenance and help desk effort for the current NC State implementation of VCL (cca 2000 blades) is of the order of 1.5 to 2 FTE's annually. Of course, VCL development and deployment of new hardware requires additional resources.

e) VCL represents a **one-stop-shopping** venue for users for a broad variety of applications. This is not only convenient for our users, but it also saves them considerable time. It also reduces the burden on college and departmental information technology staff.

f) VCL allows very **easy incorporation of new resources**, both undifferentiated and differentiated. After the hardware has been appropriately configured and connected to the network, adding it to the VCL pool is matter of several clicks on the VCL service pages [9].

g) VCL **distributes the burden of image creation and management** – often the most labor intensive part of centralized information technology management. In the case of NC State VCL implementation many of the images are created by teaching assistants, professors, and staff in different colleges and departments, while the VCL staff worry primarily about a dozen or so base-line images out of a pool of about 600 (of which about 120 to 180 are typically active in a semester).

h) VCL image methodology allows **isolation of applications** from each other, even providing the ability to allow multiple versions of the same application to be available – greatly reducing the systems administration and support effort. VCL architecture provides increased **security** and application **integrity,** again reducing the administration and security management costs.

In VCL scalability is achieved through a combination of multi-site multi-user service hosting, operating system and application virtualization using open source and commercial products, and both time and CPU multiplexing and load balancing. In our experience, students use VCL to augment their laptop and desktop capabilities by accessing any of the environments they may not have. Typically, they are allowed reservations that last 1-4 hours in one session. However, advanced users of VCL may be allowed, and do request, long-term renewable reservations. The latter scenario is primarily used for resources owned by research projects or for special case use. For example, the average duration of a VCL reservations in the Spring 2008 semester was about 1.5 hours. At any one time, about 20-25% of the on-campus VCL computational resources were in the non-HPC category while the rest were in the HPC category. The growth of VCL usage is shown in Figure 5.

4.1.1 "Green"

One other important characteristic of VCL is that it is "green". Most of its hardware is power conscious. However an even more important factor is that VCL was designed to collect sufficient amount of meta-data that it can also be pro-actively "green."

Figure 5a shows the diurnal Fall 2008 remote usage of VCL resources (non-HPC services) based on automatically collected VCL statistics. We see that the resources are underutilized in roughly the 10 pm to 10 am time-frame. In the case of physical computing laboratories most of the laboratories are closed in the 10 pm to 8 am time-frame and (in theory) computers go into an energy-saving mode. However, in the case of VCL resources (data-center based and remotely accessible) servers are not powered down and they can be used for other types of support. Figure 5b illustrates variation in the daily usage totals for Fall 2008. There are prolonged periods of time, such as during fall and spring breaks, and during Xmass or summer holidays, during which non-HPC VCL resources are less utilized. This suggests definite opportunities to either save power (e.g., by shutting down certain percentage of the equipment during low utilization periods) or to utilize it in a different way. While modern compuational and storage equipment does come in with energy-awarness and one can program it to follow energy-saving cycles, it is not always clear that this would be the best way to manage idle capacity.

An alternative is to allow secondary applications to "scavenge" computational cycles that are not being used by the primary application. There are a number of solutions in that space. The range from classical batch processing in combination with real-time

interactive processing, to distributed cycle-scavenge solutions such as Condor and Boinc (IBM World Wide Grid). These solutions do not reduce power consumption, but do increase utilization of computational resources by sharing them among different users. Of course, the key question in this context is whether such resource sharing actually results in a reduction of cost per computational service unit.

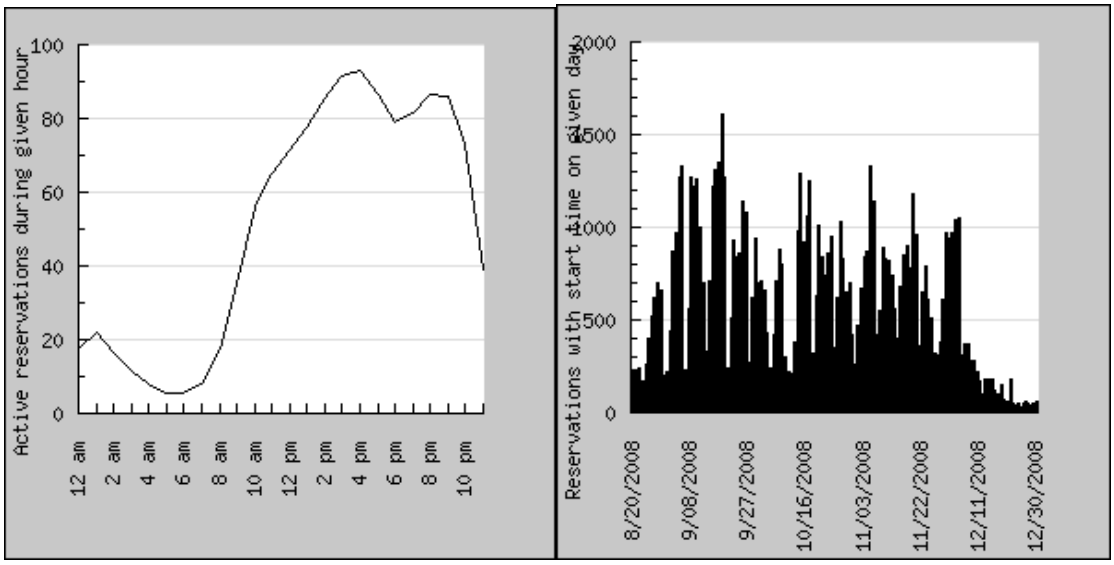

Figure 5. a) Left - Daily cycle for concurrent reservations averaged in the Fall 2008. b) Right – Total number of daily reservations in the Fall 2008.

In some situations cycle-scavenging may not really be an option. For example, when the primary application is Windows based and the secondary application is Linux based, cycle-scavenging is not the right soluton. However, if the idle Windows-based resources can be rapidly reconfigured to present to the alternative application an Linux-based platform, then this becomes a viable way of increasing resource utilization. Virtualization technologies offer this opportunity (e.g., VMware, Xen, KVM) by allowing multiple operating systems to co-exist on the same physical platform and by allowing rapid migration of idle guest operating systems. An alternative is re-loading of “bare-metal” with a new environment that the secondary applications can use.

In a global setting, there are many more opportunities to back-fill and fully utilize resources provided that the applications and customer expectations conform to the asynchronous model. It is more challenging to do that if real-time and sychronous expectations exist on either the part of end-users or the applications. For example, a user of a web-based shopping services may have expectations about how fast the transactions the user is making and observing may need to occur. If a typical user abandons transactions that on the average take longer than 7 seconds, then one has to ensure that processing and data transfer delays between the data center where the transactions are being processed and the end-user terminal do not take that long.

5. Clouds

Definitions of what constitutes a "cloud" vary [e.g., see 31, 30, 9]. However, everyone seems to agree that "clouds" and "cloud computing" is the next natural step in the evolution of on-demand information technology services and products. "Cloud computing" embraces cyberinfrastructure [21] and builds upon decades of research in virtualization, distributed computing, grid computing," utility computing, and more recently networking, web and software services [e.g., 18, 15, 22, 13, 14, 25, 28, 19, 29, 2] The term became „popular" sometime in October 2007 when IBM and Google announced a collaboration in that domain [e.g., 26, 23]. This was followed by IBM's announcement of the „Blue Cloud" effort [e.g., 24]. Since then, everyone is talking about „Cloud Computing." Of course, there also is the inevitable Wikipedia entry [27]. "Cloud computing" implies a service oriented architecture, reduced information technology overhead for the end-user, greater flexibility, reduced total cost of ownership, on-demand services and many other things [9]. It is also obvious that, to a large extent, cloud computing will be based on virtualized resources.

While today "clouds" are still relatively explicit entities growing from existing technologies, in the future they are expected to become much more invisible from the perspective of an end-user. "Cloud" resources will become available on-demand and seamlessly to a user with appropriate credentials, and will automatically augment their computational and data processing requests.

It is interesting to see how VCL can be used to construct different types of "virtual clouds" as well as cloud services on-demand. VCL already is service-oriented and offers on-demand augmentation of end-user computational abilities in both non-HPC and HPC categories. In that sense, it is already a "cloud" solutions. While VCL already has an API that allows remote reservation of the resources, today it is almost as a rule used in the manual mode, i.e., resource requests and reservations are made manually by the end-users and administrators. However, central to the ability of the VCL to take the next step in this process, i.e., provide resources in aggregates that are needed, is an extension of the VCL Image – the so called VCL Environment.

Image was discussed earlier, and typically it is a stand-alone entity that can be loaded on any number of available computational platforms – real of virtual – singly or in groups. If those images recognize each other (as VCL "cloud" images do), or register with some orchestration and management node, they may start forming networks of collaborating applications and resources. Sometimes, we may wish to always have several of the applications made available jointly. For example, a web server, a database server and an application server, or one may wish to have ten Fedora 9 loosely coupled nodes that would be used to test a workflow, along with a Kepler workflow manager image. Also, we may also wish to have those resources loaded on particular hardware, or a group of platforms. Here is how that is done.

An "Environment" is a collection of images loaded together on one or more blades or platforms and made available as a group. An Environment has a parent image and one or more child images. Whenever a request is made to load the parent image (which is the one that appears on the request menu), VCL will also load all the children images. Where the images are loaded, and how tight the coupling is among the images, depends on how they were constructed.

For example, a Web server, a database server, and a visualization application server could be offered to the user as "bundle". Parent image could be called "Web-Services Course" and would contain any one of the principal applications, e.g., the Web server. The other images would be attached to this principal image at the Environment creation time. When user reserves such an environment or "composite image" VCL reserves these three resources as a group.

When a user has the right to create an image, that user typically starts with a "NoApp" or base-line image (e.g., Win XP or Linux) and extends it with his/her applications. Similarly, when a user constructs composite images (aggregates of two or more images), the user extends service capabilities of the basic image and then attaches to that image additional (and already constructed) images. Of course, child images do not have to be identical. When an Environment is requested, the reservation is made for x real or virtual resources, depending on how the individual images were constructed. That is, if the baseline image is of the bare-metal load type, then bare-metal load occurs. If it is a VMware image, it is loaded onto the appropriate VMware server. Where this aggregate is physically loaded (location) onto the actual hardware also depends on how the image was mapped onto the underlying hardware groups. All images could be in the same chassis (if this is desired), or they could be distributed. Often images are kept together and they have fixed IP numbers. In general, the latter may not be the case. Environment can be used to map directly or indirectly on any number of hardware resources, including Blue Cloud ensembles. It is important to note that it is the responsibility of whoever constructs the images to ensure that once loaded, this small "virtual cloud" can work as it is envisioned by building appropriate discovery and communication agents and applications into this image aggregate.

When VCL images are equipped with software that allows formation of overlay networks and clouds (e.g., based on Globus, Hadoop, or Condor), then those environments can operate as parts of Globus grids, Hadoop-based clouds/services, Condor groups, or as members of any other type of clouds and services.

In fact, how a "virtual cloud" is implement will very much depend on the communication needs and set-up of its components. If the components need to be tightly coupled and the entry point for a user can be centralized, the HPC configuration of the VCL may be better suited than an Environment (or image composites). In that case, each of the HPC-mode resources (e.g., blades) would be loaded with an appropriate client image, along with one or more coordinating nodes.

The third option is the use of existing services. VCL can (and does) communicate with any number of resources that run *vclclient* daemon or agent. An example are our physical computing laboratories. That agent, along with an availability schedule that is already part of the VCL management offering, can be used to opt-in such remote resources when they become available. Similarly, instead of *vclclient*, VCL could communicate with any service that provide needed information about the end-resource and allows some basic resource preparation and scheduling tasks.

For example, there is nothing to prevent formation of a z-Cloud, a grouping of System z LPARS that would be accessed through a specialized VCL login node image. LPARS could be assigned as public laboratory resources where users do not have administrative privileges, and their creation and administrative management would be left to System z

administrators. In fact, we already do have VCL images that can access Marist College teaching System z machines.

Forming heterogeneous "clouds" – both public and private, or joining other "clouds", is a natural function for VCL since one can construct Images, Environments and/or HPC aggregates that offer different resources and different philosophies, and so long as they are appropriately managed and orchestrated, the diverse nature of these components is not a problem

A step further is to form workflow-based environments where, for example, a Kepler-based [12, 20] workflow management image that controls one or more System z LPARS, a cluster of Cell processors, and a cluster of classical blades. Kepler is a scientific workflow management environment that can be used to manage large-scale scientific workflows that span heterogenous environments, from supercomputers to personal workstations, to analytics clusters [19, 20]. It is the environment that we are in the process of adapting for construction of heterogenous computational "clouds" that include traditional clusters, cell-processors and IBM System z resources. VCL tools that can be used to form these composite environments are the group reservation functionality, the HPC cluster functionality, and the Environment (parent-child) functionality. VCL can transform into and support any type of "virtual cloud" so long as images with the appropriate environment manager and communications are provided by the image designers and implementers.

6. Summary and Plans

Virtual Computing Laboratory (VCL) is a "cloud" computing solution that can be used to implement an extremely flexible and cost-effective way of delivering a wide range of computational services – from single "seats," to groups and clusters (or sub-clouds) of real and virtual servers, to high-performance computing solutions. Its key characteristics are simplicity, flexibility, scalability, and modular nature that allows substitution and extension of different underlying components – from operating systems or hypervisors, resource management solutions and schedulers, to the underlying hardware resources. VCL has been a production solution at NC State for over four years, and is currently poised to extend to a number of other universities.

However, as with all forward looking products, there remain a number of challenges and future development directions. Our research and development plans include a) virtualization variety (currently VCL integrates support for VMWare, we are in the process of extending this to Xen and possibly KVM), b) pro-active and speculative scheduling (currently, our scheduling is primarily based on pre-loading based on fixed operational profile, and on last-served algorithms), c) fully automated image construction (although image creation process and some of the updating of image applications is automated to some extent, a considerable number of manual steps are still present), d) government and military-level security options (while VCL has an excellent security record and its security mechanisms are more than sufficient for use in academic environments, we are in the process of hardening VCL to bring it into compliance with government and possibly military standards), e) increased performance (while the performance of VCL is adequate, we are working on improving image load times through the use of different technologies, including network-based LUN booting), f) seamless

resource sharing (one of the open goals is making VCL resource requests and interactions more invisible from the perspective of the end-user – this may be particularly important in the context of K-12 deployment where end-user interface has to be very intuitive and very seamless; in the similar vein, we would like to be able to automatically and seamlessly share excess resources with other VCL installations to buffer capacity spikes, or to manage emergencies), g) interacting to other cloud solutions is one of the immediate priorities; we are currently actively working on defining APIs and integration of VCL with IBM's Blue Cloud resources; other "clouds" will follow.

Of special and active interest is making VCL facilities and paradigm available to North Carolina K-12 and higher education communities. We believe that the "cloud" paradigm and the VCL implementation represent a true paradigm shift in both access and content delivery to the broad base of educational community and are actively working, with support of a number of agencies, on making that happen.

7. Acknowledgments

The authors would like to thank all NC State staff, faculty and students who are involved with VCL operations, development and research, and all our external collaborators who are researching or operating VCL pilots and solutions. We would also like to thank reviewers for very insightful and helpful comments that have helped us improve this paper.

This work is supported in part by NC State University, the State of North Carolina, NCCCS, IBM Corp., NetApp, Cisco, and DOE grant DE-FC02-ER25809.

8. References

1. M. Vouk, "Support of Student Learning in a Technology-rich Environment: *Remote Application Access for Specialized and Collaborative Software*," IBM SUR Proposal, February 2003.,
2. Sam Averitt, Mladen Vouk, Henry Schaffer, Eric Sills, and Gary Howell, "Support for Distributed Scalable High Performance and Grid-based Computing and Problem Solving with Emphasis in Science and Engineering," a UNCGA grant proposal submitted by NC State University to UNC General Administration, Raleigh, February 2004.
3. Sam Averitt and Mladen Vouk, "The Digital Campus utilizing Virtual Computing Environment," SUGI31, San Franciso, CA, March 26-29, 2006 (Invited Talk)
4. Mladen Vouk, "Automation of Large-scale Network-Based Scientific Workflows," Microsoft eScience Workshop, The Johns Hopkins University, Baltimore, Maryland, October 13 - 15, 2006, p21
5. Peeler, A., S. Averitt, W. Baudoin, M.l Bugaev, G. Howell, E. Sills, H. Schaffer, J. Thompson, M. Vouk. Virtual Computing Laboratory at NCSU. *UNC CAUSE 2006* (http://www.ecu.edu/cause06/), Nov 2006.
6. Henry Schaffer, Sarah Stein, Aaron Peeler, Samuel Averitt, Mladen Vouk, Eric Sills, "Connecting to High-End Software Applications," UNC Teaching and Learning with Technology Conference, Raleigh, NC, March 21-23, 2007.
7. Sam Averitt, Michale Bugaev, Aaron Peeler, Henry Schaffer, Eric Sills, Sarah Stein, Josh Thompson, Mladen Vouk, "The Virtual Computing Laboratory," Proceedings of the International Conference on Virtual Computing Initiative, May 7-8, 2007, IBM Corp., Research Triangle Park, NC, pp. 1-16.
8. Mladen Vouk, Sam Averitt, Michael Bugaev, Andy Kurth, Aaron Peeler, Andy Rindos, Henry Shaffer, Eric Sills, Sarah Stein, Josh Thompson "'Powered by VCL' - Using Virtual Computing Laboratory (VCL) Technology to Power Cloud Computing ." Proceedings of the 2nd International Conference on Virtual Computing (ICVCI), 15-16 May, 2008, RTP, NC, pp 1-10

9. Mladen Vouk, "Cloud Computing – Issues, Research and Implementations," in the Proceedings of the 30th International Symposium on Information Technology Interfaces (ITI) 2008, June 23-26, Cavtat, Croatia, to appear.
10. I. Crnkovic and M. Larsson (editors), *Building Reliable Component-Based Software Systems*, Artech House Publishers, ISBN 1-58053-327-2, 2002, http://www.idt.mdh.se/cbse-book/
11. CCA, http://www.cca-forum.org/, accessed February 2006
12. B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger-Frank, M. Jones, E. Lee, J. Tao, Y. Zhao, "Scientific Workflow Management and the Kepler System," Concurrency and Computation: Practice & Experience, Special Issue on Workflow in Grid Systems, Volume 18 , Issue 10 (August 2006), Pages: 1039 – 1065, 2006.
13. Globus: http://www.globus.org/
14. Hadoop: http://hadoop.apache.org/core/
15. Condor: http://www.cs.wisc.edu/condor/
16. IBM, "IBM Introduces Ready-to-Use Cloud Computing," Nov 15, 2007, http://www-03.ibm.com/press/us/en/pressrelease/22613.wss
17. IBM, "IBM, EU Launch Research Initiative for Cloud Computing," in Grid today, February 5, 2008, http://www.gridtoday.com/grid/2102294.html,
18. Amazon Elastic Compute Cloud: http://www.amazon.com/gp/browse.html?node=201590011
19. The NetApp Kilo-Client: http://partners.netapp.com/go/techontap/tot-march2006/0306tot_kilo.html
20. M. A. Vouk, I. Altintas R. Barreto, J. Blondin, Z.Cheng, T. Critchlow, A. Khan, S. Klasky, J. Ligon, B. Ludaescher, P. A. Mouallem, S. Parker, N. Podhorszki, A. Shoshani, C. Silva, " Automation of Network-Based Scientific Workflows," Proc. of the IFIP WoCo 9 on Grid-based Problem Solving Environemnts: Implications for Development and Deployment of Numerical Software, IFIP WG 2.5 on Numerical Software, Prescott, AZ, 2006, printed in IFIP, Vol 239, "Grid-Based Problem Solving Environments, eds. Gaffney PW and Pool JCT (Boston: Springer), pp. 35-61, 2007.
21. D.E. Atkins et al., "Revolutionizing Science and Engineering Through Cyberinfrastructure: Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure," NSF, Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure, January 2003, http://www.nsf.gov/od/oci/reports/atkins.pdf
22. I. Foster, *The Grid: Blueprint for a New Computing Infrastructure*, 2nd Edition, Morgan Kaufmann, 2004. ISBN: 1-55860-933-4.
23. IBM, "Google and IBM Announced University Initiative to Address Internet-Scale Computing Challenges," October 8, 2007, http://www-03.ibm.com/press/us/en/pressrelease/22414.wss
24. IBM, "IBM Introduces Ready-to-Use Cloud Computing," http://www-03.ibm.com/press/us/en/pressrelease/22613.wss, November 15, 2007.
25. IBM, "North Carolina State University and IBM help bridge digital divide in North Carolina and beyond," May 7, 2007, http://www-03.ibm.com/industries/education/doc/content/news/pressrelease/2494970110.html
26. S. Lohr, "Google and I.B.M. Join in 'Cloud Computing' Research," October 8, 2007, http://www.nytimes.com/2007/10/08/technology/08cloud.html?_r=1&ei=5088&en=92a8c77c354521ba&ex=1349582400&oref=slogin&partner=rssnyt&emc=rss&pagewanted=print
27. Wikipedia, "Cloud Computing," http://en.wikipedia.org/wiki/Cloud_computing, May 2008
28. E. Naone, "Computer in the Cloud," Technology, Review, MIT, Sept 18, 2007, http://www.technologyreview.com/printer_friendly_article.aspx?id=19397

29. J. Reimer, “Dreaming in the “Cloud” with the XIOS web operating system,” April, 8, 2007, http://arstechnica.com/news.ars/post/20070408-dreaming-in-the-cloud-with-the-xios-web-operating-system.html
30. OGF25, http://www.ogf.org/OGF25/index.php
31. Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica, Matei Zaharia, “Above the Clouds: A Berkeley View of Cloud Computing,” Electrical Engineering and Computer Sciences, University of California at Berkeley, Technical Report No. UCB/EECS-2009-28, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html, February 10, 2009