Establishing Business Oriented System Infrastructure Development Life Cycle for Enterprise Systems

Min Jiang, Chu Jong, Paul Poppell, Keshab Budhathoky, Ryan Hull
School of Information Technology, Illinois State University
Normal, IL 61790, U.S.A
{mjiang, cjong, phpoppe, kbudhat, rlhull}@ilstu.edu

Abstract – *Service oriented organizations rely on resilient computing systems to assist them not only in automating their business but also extending their operation hours to 24 by 7. Two major categories of their computing systems are software and hardware. Many researchers have worked on the software development which emphasizes improving the performance of business processes on the application level. However, without proper supporting hardware infrastructure the effectiveness of these research works is reduced significantly. A proper hardware infrastructure design should fully support the requirements for both system software and applications. The requirements are derived from business objectives. Here we present our approach of system hardware integration design, system infrastructure development life cycle. We applied the this methodology to the two tasks, "Hardware and System Software Installation and Configuration" and "Building a Networking and Supporting Environment", which are part of ongoing research and education infrastructure development in the Enterprise Computing Systems program at the School of Information Technology at Illinois State University.*

Keywords: infrastructure, SIDLC, enterprise, computing

1 Introduction

An enterprise computing system is a set of computer technologies (hardware, software, and practices) used in integrated large scale systems. They are made up of a group of computational entities, including mainframes, servers, and peripheral devices which are interconnected by a network forming a virtual centralized computing facility. These integrated computer systems are widely used by service oriented enterprises for their business operations.

Most service oriented organizations providing 24 by 7 structures rely on resilient computing systems to allow clients to access their services in a convenient, secure, and timely fashion. Constructing a computing system to support such organizations requires collaborative efforts in many areas. These areas include: hardware infrastructure, application development, system performance, software integration, security setup, recovery strategy, and capacity planning. A well developed enterprise computing system should fully utilize the system capacity to achieve the business objectives. The study of system infrastructure design is critical to achieve high resource utilization in a dynamically evolving environment.

Many researchers focus infrastructure development on developing software applications that automate business processes [1, 2]. However, underlying hardware resource allocation, system software configuration, and network design are as important as software applications. Performances of applications are restricted by the capacity and power of underlying hardware infrastructure. The resources of underlying infrastructure should be carefully selected, allocated and configured to support the applications running on the upper layer. In this paper, we present our System Infrastructure Development Life Cycle (SIDLC), a methodology of building hardware, system software, and network environment for enterprise computing systems. Applying the SIDLC methodology, we are developing an infrastructure for both research and education at the School of Information Technology (ITK) at Illinois State University (ISU). Our results indicate that the SIDLC can also be used by other organizations to develop their enterprise computing systems infrastructure. In the following sections, we will present the methodology of the SIDLC research work followed by two implementations, "Hardware and System Software Installation and Configuration" and "Building a Networking and Supporting Environment". These implementations are two major tasks of enhancing research and education environment for the Enterprise Computing Systems (ECS) program at ITK.

2 The Concept SIDLC

SIDLC is composed of six major phases: requirement gathering, analysis, design, testing, implementation, and maintenance. It is based on waterfall model with the proper feedback loop immediately following each phase. Outcomes of each phase are analyzed and the results are used to enhance new revisions. Document control and change management system is important to maintain the infrastructure design in a consistent state. A central repository is used to keep a log of all changes and record activities for each phase. Figure 1 shows the major design flow of the SIDLC.

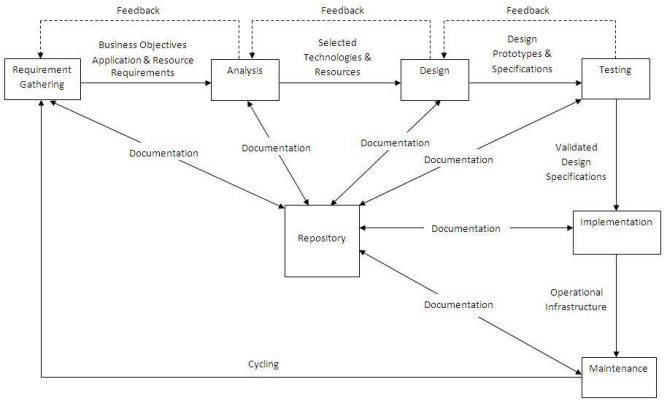

Figure 1. Process View of SIDLC

In the following sub-sections, we will discuss each phase of the SIDLC in detail.

2.1 Requirement Gathering

SIDLC starts with gathering information which includes: business objectives, applications and resource requirements, policies, security, and other business specific areas. A list of applications that will be running and their profiles should be collected. An application profile may include the application type, number of users, user locations, operation hours, response time, protocol types, security, availability, scalability, and interactions among applications. Our research concentrates on the hardware, system software, and network environment development. Application analysts will perform selecting, developing, and integrating applications. However, system infrastructure developers need to work with application analysts in regard to the application selection and development process. A suitable system infrastructure for the selected/developed applications requires two way communications, which is very important for achieving the business objectives. To successfully initiate resource allocation, collection of resource requirements for each application from application owners is needed.

2.2 Analysis

After collecting the requirements from business and applications, different technologies and resources (hardware and software) are analyzed to find out which is the best for the requirements and the selected applications. The requirements are categorized into criteria for technology and resource selection. Based on their impacts to business objectives, the criteria are assigned different weights. Each technology and resource is given a score for each criterion. From the assigned weights, a total score is calculated. Technologies and resources that score the highest are selected for infrastructure development. However, other factors (e.g. time and/or geographic difference) may change the weight of each criterion. A multi-dimensional model is used to select the most suitable technologies and resources for different situation. This multi-dimensional analytical model can be normalized with the following formula.

$$Tws = \sum_{i=1}^{n} W(t)i * Si$$

Tws: total score
W: weight for each criterion
S: score of technology or resource
n: number of criteria
t: factor

This model can be extended by setting more parameters in W() function when more affecting factors need to be considered (e.g. W(t, g)). Weights for each criterion change as time or geographical factors change. An example of using multi-dimensional analytical model to analyze candidate technologies and resources is presented in Section 3.

An important criterion that has to be considered is the cost. Based on different situations, organizations need to justify the trade-off between the best suitable resources and the cost. Technologies and resources may perform better to support an organization's business, however if the cost cannot be afforded by the organization then another choice needs to be made. So, the goal of the analysis phase is to make the best decision based on not only selecting the most efficient technologies and resources to support application performance but also the overall cost that organization can afford in a particular environment. Cost and other problems discovered in this phase provide feedback for the requirement gathering phase, allowing for adjustments.

2.3 Design

The selected technologies and resources are the basic components to initiate the system infrastructure design. Based on the analysis result, hardware resources are either partitioned, or grouped, into logical blocks, which means resources are assigned to each logical block according to their purposes, their performance requirements for the system software, and applications to run. When assigning resources, designers can reference the design of their existing infrastructure and historical documentations in their repository. If possible, designers can also refer to the design specification of infrastructure in other organizations as a starting point. With installing system software on each logical block, a platform for applications is formed. Based on the interactions among applications, internal and external networks are designed with consideration of system software and application performance. Each component of the infrastructure is configured based on its specific purpose and requirement. Then, each component is modeled and evaluated either by software or in a physical environment. After individual component modeling, integration modeling and evaluation is performed. Problems found in this phase may need to loop back to the analysis phase, or possibly the requirement gathering phase. Problems found in the beginning phases are more efficiently and economically corrected, than if discovered in later phases. Hence, the modeling and evaluating processes need to be carefully conducted and repeated until all system and network performance and other requirements are satisfied.

With the emphasis on cost reduction for IT infrastructure, dynamic resource assignment and system consolidation are two important aspects in design. System infrastructure should have the flexibility to assign resources dynamically. This enables requested resources to be reclaimed after the request is completed, thus improving the resource utilization. Flexibility also includes dynamic workload balancing and template-based provisioning. Dynamic workload balancing distributes workloads dynamically to different platforms so that customer service requirements are always satisfied even during the peak hours. Template-based provisioning builds a common platform. When a request comes, a new platform can be created based on the template. This increases productivity and decreases error by removing manual steps. More importantly, the service delivery time is lowered from days to minutes. System consolidation puts different servers running on different physical machines into one physical box. It helps reduce cost of server hardware and software, power utilization, space occupancy, and other expenses. With virtualization technology, physical resources can be more effectively shared among platforms. Thus it helps implement dynamic resource assignment and system consolidation.

2.4 Testing

Applications selected by organization are tested on a prototype. This includes installation, configuration, performance testing, and interactions among applications. Installation and configuration of applications are tested on their platforms according to the application analysts/developers' design specifications. Then, each application is tested individually in its own environment. After each individual application is tested until the performance being accepted, they would be integrated together to conduct a testing at the system level. At this level, the entire system performance is tested including collaboration among applications, and interaction between applications and the platforms. Test results provide feedback to the design phase, the analysis phase and/or the requirement gathering phase. This process is repeated until test results satisfy the requirements. At last, overall integration testing including operating procedures, supporting structures, troubleshooting and problem solving processes is performed. Any failure or error detected in this step should be investigated and fixed before implementation.

2.5 Implementation

For the implementation phase, we need to consider two different cases – implementation with and without an existing production system. In the case with no existing production system, infrastructure can be implemented according to the validated design specification. For an existing production system, it is more desirable to implement new infrastructure in incremental fashion. In this way, implementation is completed step by step, trying to avoid a serious unexpected impact to the existing production environment. Errors at this step should be fixed and analyzed, checking for similar or related errors not detected in earlier phases. After the new infrastructure is successfully implemented, it should be run in parallel with the existing production system for a certain period of time. This will build a confidence level in the new system. During this time, the old production system will serve as a backup system.

2.6 Maintenance

New infrastructure needs to be maintained on ongoing basis and accommodate changes in the environment. In a big organization, business is constantly changing. As a result, new services and applications need to be frequently added or removed in the production environment. The number of users using the system will also change with business. In order to make all the applications perform in an acceptable way while supporting users' requirements, there needs to always be adequate resources to satisfy these requirements. This requires a quality system while retaining a network capacity planning strategy.

Systems and network performances should be constantly monitored. There are many tools that can monitor and display the system and network performances and resource utilization in real time. These tools can also save the information in storage for future reference. History data can be analyzed to find out the baseline and time to time resource utilization [3]. These results are then used in capacity planning for new applications and services or additional users.

Each request to add a new application into the production environment should go through an investigation process. This investigation will collect information and requirements of the new application. It includes what kind of system, database, and network environment the application needs, how many users will use this application, where those users are located, and performance requirements of the application. Based on the analysis of the information, the platform best suitable for the application will be decided. The application is then tested in a testing environment separated from the production environment. Based on the baseline and time to time resource utilization information derived from history data, the production environment can be simulated. Now, the new application is tested on a simulated environment to predict the performance and impact new application will bring to the network and other systems. This will help decide if more resource allocation or capacity upgrade for the systems and network are needed to support the changed resource requirement. Changes on the number of users or any other upgrade or change to the production system environment should follow a similar process to ensure that supply satisfies the demand. Changes on system environment do not always increase. Sometimes, applications or services need to be removed from the system environment. This may result in lower resource utilization of the system and network. In this case, systems can be consolidated together, with excessive resources being shut down or downgraded to avoid wasting resources.

All the processes from requirement gathering to implementation should be properly documented in organization's repository. Also, with maintenance and changes to the environment, documents should be updated accordingly. Documents are a good resource for troubleshooting potential errors that may occur in the future. This also provides a great reference for modification to the existing infrastructure or new infrastructure development.

3 Research/Education Infrastructure

The following SIDLC implementation tasks were performed, "Hardware and System Software Installation and Configuration" and "Building a Networking and Supporting Environment". These tasks are major part of the research and education environment infrastructure development for the ECS program.

3.1 Hardware and Software Infrastructure

One factor that may determine the enterprise computing system's requirements are the users. The ITK department at ISU is much smaller than many other organizations that develop enterprise computing systems. The primary users of this system will be students who take ECS courses. Students taking other courses such as database processing and external data structures may also use the system.

Our infrastructure development project is intended to develop different computing systems to provide a better environment for research and education in ECS program. Multiple activities for research, education and development will be hosted in this environment. This not only provides a teaching and learning environment for ECS and other related courses, but also enables us to create research and development facilities for students and faculty to conduct their works such as performance testing. In addition, it will be the main platform for collaborative activities with other institutions.

3.1.1 Hardware Installation and Configuration

The infrastructure development work is mainly around the IBM z890 mainframe machine, which was loaned by IBM to ISU for educational purposes. This machine is equipped with 32 GB of main memory, four processors, and a set of I/O devices. In addition, this machine came with a DS6000 DASD storage unit with 9.6TB capacity.

Since the mainframe machine will be used for multiple purposes such as research, education, development, and testing, it was decided to use virtualization technology. This technology allows us to divide the machine into two separate logical partitions (LPARs). The first named PIKEY is for testing which can be reinstalled and reconfigured with different operating systems as new testing needs to be conducted. The second named ZEWOK is for production systems on which research, education, and development works will be hosted. This configuration allows users to perform a wide variety of testing while not posing a safety risks to the other users. The separation of partitions would be highly advisable to any organization that requires both testing and production environment. By using separate partitions, organizations can ensure testing systems do not interfere with production systems. Another advantage of using more than one partition is that resources can be reallocated from a testing system to a production system as needed. For example, during overnight batch processing, this can be useful since many operators who work on the testing system will be gone and those resources will be unused [4].

CPU and Memory

The configuration of PIKEY will need to be changed as the testing requirements change. During the installation, ZEWOK was given 16GB memory with one dedicated CPU. PIKEY was assigned 6GB memory with also one dedicated CPU. The remaining two CPUs are shared by two LPARs, and 10GB memory is assigned as extended memory. Two LPARs also have access to a cryptographic coprocessor and all available channels. Since there was no other information or statistics available for the workloads that the system would have, for resource allocation we referred to the specification of another z890 mainframe machine, which is currently running as production system in the Administration Information System (AIS) department of ISU. But, this configuration is just a starting point. After testing is completed, we will be able to assign resources more accurately based on the workloads of the system.

DASD

For our system, the DASD was reinitialized at RAID5 with 7+1 configuration emulating 3390 mod 9. The need for high performance DASD service is not that large. Considering the huge volume of information for future research, education, and development works, we chose a configuration suited for high storage capacity at the expense of I/O speed. If an organization was in need of I/O speeds, 3390 mod 3 would be a better choice. If PIKEY had a project that requires a different configuration, one RAID configuration could be rebuilt to match the project needs. Our configuration of RAID5 7+1 gave a high storage capacity with tolerable performance. When considering the performance versus capacity, we tried to select the best option from RAID 0-5.

3.1.2 System Software Selection and Installation

As we discussed before, ZEWOK will host different activities related to research, education, and development. All these activities will be going on simultaneously, and they need an independent environment so they will not affect each other. In addition, these activities may run applications that require different architectures.

z/VM on ZEWOK

To satisfy the requirements, we virtualized ZEWOK using IBM's z/VM operating system. z/VM allows the sharing of the mainframe's physical resources such as disk, memory,

network adapters and CPUs. These resources are managed by a hypervisor, called Control Program (CP) [5]. When the user logs onto a z/VM's guest account, the hypervisor creates a virtual machine which can run one of many different operating systems. So, z/VM can host many different virtual machines concurrently running different operating systems as guests on top of it, and manage the physical resources for sharing between the guests.

z/VM also allows for great flexibility and scalability, which is very important to accommodate the classroom growth and changing requirements for different activities. Resources and new systems can be dynamically added or removed as needed. z/VM can be compared to a parent feeding the children who are like the guest operating systems. The parent has limited amount of food to give to the children who may require different amounts of food based on their activity levels. z/VM balances the resource allocation among guest operating systems and tries to satisfy everybody's needs.

Linux on z/VM

When considering the guest operating systems to run on z/VM, Linux is a good option because it provides low cost, high reliability and a wide variety of customization [6]. To illustrate how Linux might be a good option for an organization running an enterprise system, we will discuss how it could be applied to our system.

Obviously, the selection of operating system will vary from organization to organization. In our infrastructure setup, we chose SuSE Enterprise Edition primarily due to time and cost. SuSE has many features that we require so it is currently the best choice. Table 1 compares several different Linux operating systems in terms of the features they provide [7, 8].

Operating system/Requirements	Red Hat Enterprise (RHEL)	SuSE Enterprise Server (SLES)	TurboLinux
Software application	Conga – cluster management software	ZenWorks- software suite that provides automated installations and updates	TurboLinux Cluster Server - similar to beowulf clusters without special software
Built-in/available security software	Security Enhanced Linux	AppArmor	Security Enhanced Linux support Trend Micro Server Protect for Linux
Software support	584 product vendors	Yast	Mongoose package manager
Product support	24/7 support	24/7 support	90 day installation
Cost	$1,349	$720	$749

Table 1. Comparison of Different Linux Operating Systems
*Note: There are many other features and options. This is a very small sample chosen arbitrarily

In the following paragraphs, we will explore how we selected the operating system that meets our requirements. To simplify the discussion, we will cover the top three.

The integration of the Linux operating system into a system's infrastructure needs to be a planned event. This should involve team meetings with representatives from all departments who will be working with the system including product vendors. Requirements gathered from the team meetings should be carefully analyzed when selecting the operating system. In order to choose the correct operating system, we used the analytical model introduced in Section 2. We categorized the requirements into several criteria. Based on their importance to our objectives, criteria are prioritized by being assigned different weights. Each candidate operating system is given a score (scaled from 0 to 100) for each criterion. Based on the weights assigned to each criterion, a total score is calculated for each candidate operating system. Since our environment is not affected by time, geographical location, or other factors, a two-dimensional model was adequate.

Criteria	Weight	Linux 1	Linux 2	Linux 3
Cost	25%	70	60	90
Supports business objectives	20%	80	90	60
Scalability	15%	80	90	80
Compatibility with other technologies	15%	80	70	90
Availability	15%	100	100	90
Security Features	10%	45	50	15
Total Score	100%	72.5	**79.5**	62.5

Table 2. Operating System Selections

As shown in table 2, the most important criterion is cost. Funding is always limited in any organization including ours. By selecting the most reasonably priced operating system with the required features, an organization can increase the amount of funding available to purchase other software.

The next most important requiremtent is how the operating system supports our objectives. Referring to the applications listed in table 1, we can see that TurboLinux Cluster Server and Conga are both used for clustered computing. At ISU, we have several courses that cover topics related to these applications, but SuSE's ZenWorks would be of more use in maintaining our guest operating systems.

Currently, security is the least important in priority, but it will soon become one of the most important criteria. At this moment, the enterprise computing system at ISU does not have many users or store critical information in it, so accidents would not be costly (usually results in a learning experience). Once the system is fully developed and more courses start using the system, security will be a very

important part of the infrastructure development. Only a limited number of users should be able to perform administrative operations. It is dangerous even to give trusted users elevated permissions because mistakes are relatively easy to make.

There are two primary security packages that most mainstream Linux distributions offer, NSA's SELinux and SuSE's AppArmor. AppArmor features an integrated security suite that is easily configured and relatively safe. SELinux is noted for being able to lock down the system securely although it is quite difficult to implement [9]. For our system, AppArmor is the best choice due to the time constraints.

Based on this two-dimensional analytical model, we could conclude that SuSE would be the best choice for our enterprise computing systems infrastructure.

Resource Requirements for Linux

Now that the operating system has been selected, we can consider the resources required for our Linux installation. Again, creating a chart is a useful way to visualize the requirements as shown in table 3 [5, 10, 11].

Application	Apache/Ant	Kuali	Oracle
Resource requirement	109MB Disk	4GB Disk 2GB Memory	9GB Disk 512 MB Memory 1,500 MB Swap
Other requirement	Java Runtime Environment >1.5	Oracle Tomcat 5.5 JRE	-

Table 3. Resource Requirements for Linux
**Note some dependencies have been left out

This simple chart was created to demonstrate the resource requirements for the Kuali financial systems which is an application we plan to port to our z system environment. Based on table 3, we should assign at least 9+4+(~5 SLES) = 18GB of disk space and allocate a minimum of 2GB memory.

Design of Initial Linux Guest

Our first prototype was from the initial Linux guest installation by an IBM engineer. The primary goal of the first installation was to verify that we could install a Linux guest by ourselves. After this was completed, we installed another Linux guest and documented the process. These steps were completed rather quickly because other users needed to begin their research. Organizations should begin by getting the higher priority systems up and running first. Later, it can be refined when further testing and analysis has been completed.

During the first guest installation, we chose to use 2 volumes based on an IBM cookbook. Later, we only used one volume (6.8GB) to perform the installation, which didn't provide that much extra space. As mentioned before, our educational systems will require a larger space so we will need between two to four volumes for one guest system.

When we start creating new guests, we plan to use an automated cloning script such as clone.sh provided by IBM. This can save time editing configuration files and fixing errors. This will actually be done during the implementation phase, but might be useful if we need to create multiple testing systems.

An important note in the Linux guest installation is that allocating just enough memory so Linux just begins to swap. This is because Linux will try and fill up any available free space with disk cache. If we select the minimal memory size, Linux will be forced to swap frequently but will not try and fill up any available free memory. Normally, this would be undesirable, since disk access is very slow. When we are using z/VM, the pages are virtual (pages are contained in memory) so paging occurs nearly as fast as memory access.

Installing First Linux Guest

The first step in the installation is to prepare minidisks for the image. After formatting the minidisks, we proceeded to edit several configuration files using z/VM's Conversational Monitoring System (CMS). Each time we updated a new configuration file, we proceeded to backup the original instance of the file. By creating backup copies, we have a restoration point if we need to perform a system recovery.

For each Linux installation, we need to specify a virtual NIC card if the system will communicate with the outside world. For our installation, there was an additional configuration file that needed to have a NICDEF (NIC card definition) statement, which was not included in the cookbook. This is a good example of how important the documentation of the actual system is.

Testing

After the prototype installations have been completed, the performance testing of the new guest can begin. For our testing purposes, we plan to use some of the testing suites provided by IBM like the Performance Toolkit. As time progresses however, we plan to develop our own testing suites that will give additional functionality.

The installation and maintenance of the educational guest operating systems has yet to take place. Hopefully, during the fall semester of 2009, some of the major installations will take place and testing can begin. As we continue our infrastructure development, we will be able to provide more examples of our installation to organizations considering developing their system infrastructures.

Documentation

One mistake that can easily be made is rushing the installation without proper documentation. The initial guest was installed in several days with the help from IBM. Some notes were taken during the installation and several more followed with the second guest installation. During the third

installation, a customized cookbook for our system was put together. Trying to remember everything that was done during the installation would be nearly impossible. This is especially true for organizations where there may be long time lapses between such processes.

Another factor that will require such documentation practices is that most organizations are continually recruiting new employees. In ISU, both these factor play an important role in why we developed specific documentation practices. Students are continuously graduating and new students are coming into the ECS program. Good documentation helps new students catch up with the information in a short period of time.

3.2 Networking and Supporting Environment

The network is considered one of the most critical resources in today's computing system [12]. Fulfilling the demands from customer and supplier is a decisive factor to the success of an organization. Mainframe based transaction processing systems need to provide high availability, security, performance, and responsiveness to their client not only for TCP/IP and SNA networks but also the combination of both. In this section, we will present how to design and setup a networking and supporting environment for the ECS research projects.

3.2.1 Requirement Gathering

Our objective is to setup an environment for both research and education activities for the ECS program. Planning, designing, testing, and implementing such environment is a challenging task especially with scalability, reliability, recoverability, and security in mind.

We started meeting with faculty and students for high level requirement gathering mainly on how to support their research, education and development. These high level requirements are used to construct our implementation requirements. We conducted several meetings with AIS staff and network administrators discussing possible ways to construct our research environment. At the mean while, faculty and students are involved to finalize the needs for their research, education and development work.

3.2.2 Analysis

The analysis phase is based on the following: 1) how many users; 2) local and remote logon; 3) open system and mainframe connectivity; 4) mainframe system remote administration; 5) open system administration; 6) repository management; 7) backup and recovery; 8) small classroom setting; 9) distance education; and 10) lab access. The results of our analysis are grouped into five categories [13].

Accessibility – users should be able to access both mainframe and open systems in the same location. Each individual will have different access right and privilege level. Remote access needs to go through secure firewall.

Availability – Administrative systems are required to be fully accessible all year around except for scheduled maintenance. System backup and recovery procedures will be enforced. Development systems will be managed in part by developers during assigned project period.

Connectivity – All systems need to be connected via local network. The mainframe system located in another building needs to be connected by a fiber channel via secure ports assigned by university's network administrator. Fixed IP addresses are used by servers for permanent connection and dynamic IP addresses are for temporary connection.

Classroom setup – Smaller number of students of advanced classes may access the system console directly. Large number of students of basic classes will need to access system resources via terminals. Distance learning classes will need to access mainframe via either the Internet or the ISU network.

Authentication/security – Mainframe access authentication and security will be using RACF. Open system authentication and security will be using encryption and LDAP.

3.2.3 Design

The conventional network design has been recreated [14]. New approaches used by SIDLC are to meet the changing requirement for the network infrastructure design. We put requirements from users and applications in early stage to avoid changes in the later stage. When designing our environment, five factors are to be considered.

1. Location
2. Users
3. Devices
4. Application
5. Activity

The Network Design Consideration

Past Approach	**New Thinking**
Locations	Users and Applications
Bandwidth	Latency
Network Silos	Holistic Strategic Plan
Generic Network Design	Plan for Application Architecture changes, Integral part of Application planning and Deployment
Corporate Network	Delivering Apps to support Business Processes

Figure 2 shows both inter and intra networking floor plan for the connections. Dedicated ports are used to secure the communications between our lab computers and the mainframe systems. Part of the plan is to set up mainframe remote administration (HMC, DASD, and console terminal) in our research lab. Another part of the plan is to allow students attending distance education class to access the MVS via the ISU network (the upper link on the left side of the figure).

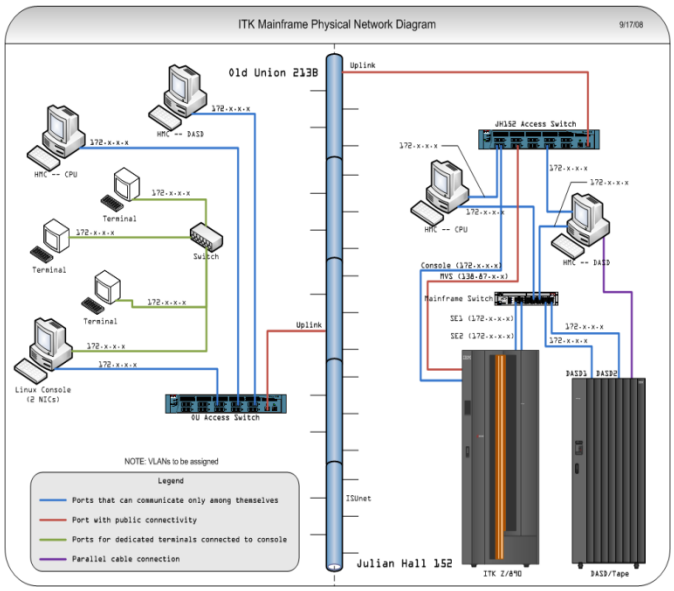

Figure 2. Inter and Intra Networking

Figure 3 shows the domain controller configuration for local and remote authentication and enforce security policy. File server with mirror provides backup and recovery ability to ensure the development will not be affected by a single failure. Print server allows users to share printers in the network. Currently both servers and domain controller are in the same computer of the local VPN. This figure also shows how multiple mainframe console sessions can be established.

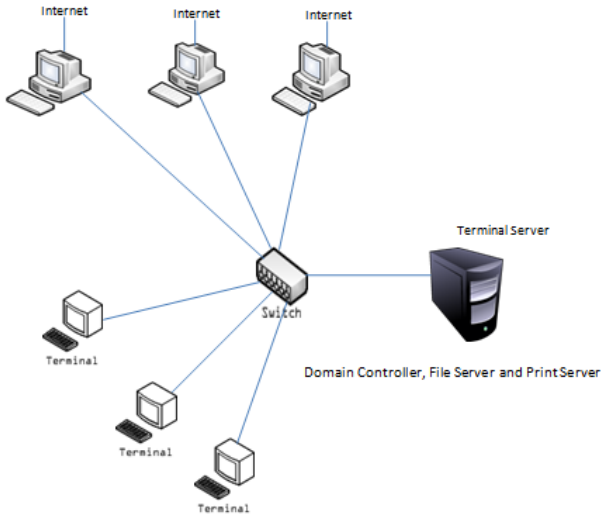

Figure 3. Networking for Research and Development

3.2.4 Testing

We plan to create a prototype in our research lab using VMware to conduct our testing prior to the implementation. Tools to be used to test the network performance, status and problems are: netstat, traceroute, ethtool –S eth0, and the main tests include: speed, latency, capacity, and congestion tests.

3.2.5 Implementation

We plan to configure the computers in the lab with different flavors of Linux operating systems. We used command line and set up configuration file manually rather than using graphical tools. They provide us with the in-depth knowledge of how it actually works and what is happening on the background while the process is running. The domain controller is configured for centralized authentication with a DHCP server providing IP addresses to client PCs in the lab. The static IP addresses are used by servers which provide permanent service for users.

3.2.6 Maintenance

Some network problems are not easily detected until they happen such as mismatched IP addresses, firewall and interface configurations [15]. Lack of connectivity and slowness of the network are the two major types of network problems. Common problems of connectivity are: a) cable mis-match; b) network interface card; c) switching port; d) power failure; e) application or server shutdown. Common problems of slowness are: a) network speed; b) NIC duplex and speed incompatibilities; c) network congestion; d) poor routing; e) bad cable; f) electrical interference; g) overloaded server; h) mis-configured DNS. All these potential problems need to be well documented, and proper troubleshooting and recovering procedures need to be established for better maintenance.

4 Summary

Building a computing system infrastructure for enterprise is a critical task. A proper system infrastructure will not only improve the system performance but also reduce the operational cost. Formalizing a methodology to assist the infrastructure construction is very challenging and we have not seen many successful cases. When we started working on the z890 mainframe system configuration, we were frustrated by not having a systematic way of building our own system. Then we began to construct an infrastructure building method for our needs. Later on, we decided to generalize our method to fit most business computing infrastructure setup. The System Infrastructure Development Life Cycle, SIDLC, presented in this paper although is still in its infancy, it started delivering useful results. Many works still need to be completed, and our plan is to continue using our implementation results to feedback to the SIDLC design until its maturity.

5 References

[1] Themistocleous, Marinos, Zahir Irani (January 2006). Towards a Methodology for the Development of Integrated IT Infrastructures. *Proceedings of the 39th Annual Hawaii International Conference on System Sciences, volume 08,* 182.

[2] Gotel, Olly, Vidya Kulkarni, Des Phal, Moniphal Say, Christelle Scharff, Thanwadee Sunetnanta (2009). Evolving

an Infrastructure for Student Global Software Development Projects: Lessons for Industry. *Proceeding of the 2nd Annual Conference on India Software Engineering Conference*, 117-126.

[3] Clewett, Annette, Dana Franklin, Ann McCown (1998). *Network Resource Planning for SAP R/3, BAAN IV, and PeopleSoft*. New York: McGraw-Hill

[4] Park, Sungsim, et al. (October 2001). *Capacity Planning for Logical Partitioning*. Retrieved May 30, 2009, from : http://www.redbooks.ibm.com/redbooks/pdfs/sg246209.pdf.

[5] McGregor, Geoff. (July 2008). *System Requirements*. Retrieved May 12, 2009, from: http://www.kuali.org/downloads/.

[6] Peckover, Lester, et al. (August 2008). *z/VM and Linux on IBM System z, The Virtualization Cookbook for SLES 10 SP2*. Retrieved May 30, 2009, from: http://www.redbooks.ibm.com/redpieces/pdfs/sg247493.pdf.

[7] Novell. (2009). *Novell ZENworks Configuration Management*. Retrieved May 30, 2009, from: http://www.novell.com/products/zenworks/configurationmanagement.

[8] IBM. *System z PR/SM*. Retrieved May 30, 2009, from: http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp?topic=/eicaz/eicazzlpar.htm.

[9] Morris, James. *Have You Driven an SELinux Lately?* Retrieved May 30, 2009, from: http://namei.org/ols- 2008-selinux-paper.pdf.

[10] Smiley, John. *Installing Oracle Database 10g Release 2 on Linux x86*. Retrieved May 12, 2009, from: http://www.oracle.com/technology/pub/articles/smiley_10gdb_install.html.

[11] (November 2008). *Apache Ant-1.7.0*. Retrieved May 30, 2009, from: http://www.linuxfromscratch.org/blfs/view/stable/general/apache-ant.html.

[12] Ebbers, Mike, Christopher Hastings, Matt Nuttall, Micky Reichenberg (August 2006). *Introduction to the New Mainframe: Networking*. IBM Redbook.

[13] Weller, Rica, Ross Clements, et al. (March 2007). *Introduction to the New Mainframe: Security*. IBM Redbook.

[14] Fabbi, Mark, Eric Paulak. (March 2009). *Five Dimensions of Network Design to Improve Performance and Save Money*. Retrieved May 26, 2009, from: http://www.gartner.com/DisplayDocument?ref=g_search&id=906112&subref=simplesearch.

[15] Linux Home Networking. (January 2008). *Simple Network Troubleshooting*. Retrieved May 24, 2009, from: http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch04_:_Simple_Network_Troubleshooting.