Migrating Infrastructure to GNU/Linux

Infinite Delta Corp, www.InfiniteDelta.com, Ty@InfiniteDelta.com
2460 Maple Valley Dr SE, Kentwood, MI 49512

April 10, 2009

Contents

Migrating Infrastructure to GNU/Linux

1 Introduction

Xyz LLC is considering migrating their infrastructure to GNU/Linux. This paper includes thoughts and issues to be addressed to help ease the transition. It focuses on paradigm differences between GNU/Linux and the systems being replaced. It emphasizes understanding and organizing around the end customer needs.

The initial intimidation of a major migration is easily dismissed by loading a test PC with a GNU/Linux. The physical installation and IT administration of GNU/Linux is easier than that needed for the systems being replaced. At one engineering/manufacturing site, a PC conversion only takes 10 minutes of a technician's time. More than a dozen non IT people have been trained to install just such systems.

Training and some frustrations are expected. Plans must be put in place to minimize frustrations, before the formal migration takes place. Support personnel with GNU/Linux familiarity are important. Long time GNU/Linux experience is not required, but new support people will need some time and practice with GNU/Linux.

Migrations become easier when people are led, not pushed into these transitions. Consider individual incentives and system improvements to help the transition. When migrations are planned, potentially huge opportunities to improve the system are possible, but bad planning and missed requirements can turn a migration into a disaster. Consider infrastructure corrections such as from a server to a distributed topologies for a more robust system. This can be a good learning curve for everyone, specially when the end users are part of the solution. This can help motivate people to consider the change.

This paper covers some migration issues Infinite Delta Corp has experienced.

1.1 Identification

This paper is identified within Infinite Delta Corp as:

```
$Id: going_linux.tex 27 2009-04-10 21:23:47Z ty $
$HeadURL: svn+ssh://InfiniteDelta/usr/local/src/drm/config/doc/going_linux.tex $
```


1.2 Acronyms

DMZ Dead Man Zones in references to networks kept away from sensitive internal networks.

ECM Electronic Capabilities Management

FSF Free Software Foundation is the core sponsor of the GNU project.

GCC GNU's Compiler Collection "All Time Most Valuable Software".

GDK GIMP Drawing Kit

GFDL GNU Free Documentation License which has some limits on distributing documentation.

GIMP GNU Image Manipulation Program

GNU GNU Not Unix a Free Software Foundation (FSF) organization http://www.gnu.org/

GPL GNU's Public License (GPL) is the corner stone to much of the free software model.

GSFDL GNU Simpler Free Documentation License is not yet released, but helps resolve issues with GFDL.

GTK GIMP Toolkit a LGPL GUI.

GUI Graphical User Interface

ID Identification

IT Information Technology, usually a department within an organization to support the computer network.

KDE K Desktop Environment

LGPL Lessor GNU's Public License (LGPL) allows linking proprietary systems to the free software.

QPL Trolltech Q Public License

QT Cross-Platform Rich Client Development Framework

RFP Request For Proposal

SSH Secure Shell

SQL Structured Query Language

X11 Also X-Windows is a network based windowing system used on most multiuser computers.

2 The free software paradigm

Several paradigm shifts become apparent when moving to the free software model. They largely change to how one does business.

2.1 From product to service

Free software has its greatest effects on the software delivery business model. As a developer there is no profit in selling free software, only in supporting and servicing it. As an end user, the free software one has installed will continue to be supported even if the originating developer is not available. For these reasons, a much more service oriented industry is evolving in the free software world. There is a general shift from products to service.

It is important to note that GNU/Linux supports running proprietary software. However, many organizations are tired of being hostage by these proprietary systems and are starting to avoid them, especially once they realize dependability and flexibility of the free software model.

2.2 From copy protections to copy encouragement

Unlike proprietary systems with their many protections to prevent copying, most GNU/Linux systems have an excellent distribution system to encourage and even automate copying. Specifically, Debian's GNU/Linux has an extremely effective download and update system.

A feature of the Debian system that Infinite Delta Corp really likes allows the creation of virtual packages which depend on other packages. Example: A virtual package "my_project" is created that is dependent on a compiler, some utilities, and specific test applications. Some time into the future, a new utility is added to the my_project dependency list. All the PCs that have "my_project" can be updated automatically and you don't even need to track which ones they are... but you could if you so wished. Virtual packages can greatly simplify custom administration.

2.3 From separately purchased development environments to included development

Most GNU/Linux distributions come with several free complete software development systems. A separately bought development system is not necessary. Many developers are pleasantly surprised as to the quality and productivity that these tools deliver. Most revolve around the GNU/GCC (GNU's Compiler Collection) - which is the standard for many platforms including GNU/Linux. GCC is heavily used, even in the aircraft Flight Safety world. No other compiler has as high a user support group nor support for more platforms. Infinite Delta Corp sincerely believes the GCC is the all time most valuable software ever produced.

2.4 From hidden source to fully accessible source code

Everyone has access to the source code. First impression for a new comer is that if there is a security vulnerability in your system, it is wide open to the public. It is important to realize that those involved with free software know this and are quick to review and close any vulnerabilities. The final system ends up being far better reviewed therefore more robust and protected then systems not open to public review.

Open source code allows reviewing source code which is a great way to learn the many methods available for doing a task. There are extensive opportunities for programmer to modify code for their own needs. Even the application configuration is usually kept in the open for users to see. Standard administrator configured protections will prevent the casual user from modifying configuration data.

2.5 From a directed path to unlimited options

The lack of direction is a pervasive impression for many new to the free software world. The huge number of varying user interfaces and desk tops are good examples. The GNU/Linux supports many desktops, often at the same time. This allows users to really customize their environment. It can also make it difficult for IT support. Choosing an initial desktop for an organization allows for some consistency in training and support. Plan on eventually supporting some variation as everyone learns what is best for them. There are many different levels of language support which may need to be considered.

The GNU programming guidelines are a good starting point to consider, especially, if you want your application supported across many platforms beyond GNU/Linux. The GNU build environment allows the applications to use some specific features. See GNU's automake and autoconf tools.

2.6 From planned obsolescence to backward compatibility

Many proprietary systems have a "planned obsolescence" business plan, which forces their customers to upgrade or change their system periodically. This can be very expensive for the customer, especially if they need to support their own system for longer periods.

The free software model prefers compatibility and tries to avoid "planned obsolescence". Much of the software written in the 1980s is fully operational on the latest systems. (This document is written using LaTeX which has been around since the mid 1980s.) This minimizes the need for constant customer retraining and unnecessary system upgrades. Note: This also means that many current distributes comes with a lot of old software not recommended for new applications or new users, but preserved for sake of backward compatibility for long time users and older systems.

Many proprietary systems intentionally avoid backward compatibility to enforce planned changes. This may be a great strategy for profit for the vendor, but it is rough on standards and the ultimate customer.

2.7 From single user to true multiuser

The biggest transition for most programmers and system administrators is the multiuser heritage of GNU/Linux. The GNU/Linux is a free software following the POSIX Unix interface, which spent much of its time in Universities. There is nothing like lots of intelligent and bored university students to test a system, including its security features. Robustness grows in this environment, with many users on at the same time, and with many developers. GNU/Linux has a strict control over running programs, protecting them from other programs. Administrators and programmers should become familiar with how protections work and some of the protection variants available.

2.8 From local user applications to network based applications

Users and developers often worked remotely on the early systems. Like today's Web interface, the GNU/Linux standard X11 graphical interface is designed to work remotely through a network interface. Hence, all applications, with a few rare exceptions, can be run remotely. No extra thought or effort needs to be placed into its design. Example: Try "rlogin -Y remote_pc", and running a graphics application, editor, anything! The X11 OpenGL support can far out perform most Web graphic applications.

The Apache web server is one of the greatest used web servers, and it comes with nearly all GNU/Linux distributions. The web is the standard for information dissemination and open information gathering. A web system often requires more effort for gathering secure information, than the standard X11 application running the Secure Shell.

The cost-to-performance of simple multiuser server application is hard to beat. It also better paves the way for a truly robust distributed architecture which is discussed later under 6.4.1 Robust Distributed System.

2.9 From proprietary licensing to free licensing versions

Management needs to be aware that each application comes with different licensing agreements. The GNU's Public License (GPL) and Lessor GNU's Public License (LGPL) are the corner stones to GNU/Linux. However there are many different licenses out there. The Debian GNU/Linux project contains many applications with many different licensing. Debian will not allow a package into their system unless it meets several strict tests allowing totally unrestricted redistribution. Even the GNU documentation does not meet Debian's strict distribution tests. Many other GNU/Linux distribution are actually based on the Debian distribution. Therefore, those distributions may have

more relaxed rules, allowing packages that Debian could not. See http://www.debian.org/intro/free and http://www.gnu.org/software/software.html.

There are no warranties with GPL software. However, warranties can be supplied by a service provider.

Like source code, most documentation also comes with the applications on GNU/Linux distributions. However, much of the GNU's documentation is under GNU's Free Documentation License:

http://www.gnu.org/licenses/licenses.html#FDL.

Because Debian's higher standards prevent GNU's documentation from being distributed with the Debian GNU/Linux, a separate link to http://www.gnu.org/manual/manual.html is required. This is expected to change when GNU Simpler Free Documentation License (GSFDL) has been approved, released, and key documents for GCC/Make/Autoconf have been moved to it.

An organization's assigned group or individual will test and approve or reject software for the organization.

These licensing may be restrictive to certain applications, so understanding the limitations is important.

3 Requirements

The key to a successful organization is the degree to which it practices the efficient utilization of its capabilities. Understanding these capabilities with their dependencies are critical for good planning. Requirements are the itemizing of those capabilities and dependences relations. Awareness of requirements and their importance must not be lost. An in depth interview with all users, even if only electronically conducted, is highly recommended.

The following sections include some obvious and some not-so-obvious system requirements.

3.1 Desktop/e-mail/web/documentation/distributions

There are many GNU/Linux distributions, e-mail clients, web servers and clients, and documentation systems. The relative merit of these systems are very subjective and need to be resolved locally. They can also have different levels of language support.

Infinite Delta Corp prefers Debian GNU/Linux distribution, but also has used the Knoppix Live/CD for PC candidate checking and diagnostics.

The IT people involved need some time on each candidate, so they support their institution's planning process. Many departments may vary within an institution. Engineering and manufacturing departments may require more IT support to handle the variations, where as the business departments often can be rubber stamped machines.

3.2 Networking and security

The GNU/Linux has widely used firewall and gateway systems. Basic "iptables" style of protection usually can be first time configured in a few hours by an IT person. The Secure Shell (ssh) remote access gives a fairly good level of security for remote users and copying data. Be sure to use SSH protocol 2 or newer.

Often companies will use hierarchy of networks to improve and simplify security and network traffic. Over-dependency on the network can also be a problem. The proliferation of file servers, web interfaces, and phone systems often puts a heavy burden on a network. The dependency on the

network also can stop an organization when the network goes down. With simple planning and minor IT work, much of the minute-to-minute dependency on the network can be eliminated. Network down time costs must be included in the plan.

Individual and server backup policies need to be planned out. A backup server can, using excess hard drive space on other systems, coordinate backups. A system recovery can be trivial with a generic install disk. Excess CPU time also can be coordinated using this scheme.

Secure data often can be eliminated from systems, thus, eliminating potential risk. Example, a payroll check writing system in a local building may only need to keep the user ID and name and last date paid, whereas the main payroll system can be safely locked away in a central building.

The Apache2 web server usually can be first time configured in a few hours by an IT person. This does not include the web content, only enabling the web server.

4 Planning

Critical IT people, managers, and programmers need to be up on GNU/Linux early as possible to help planning. Long before decisions can be made, critical people need to see and touch some of the different distributions. They will quickly be made familiar with the issues they face as they start to plan the migration. IT personal will have a good idea as to what is needed by loading and configuring test systems and test networks.

There is a tendency to fall into the trap associated with single user PC architectures, when there are many options available.

5 Implementation

Some critical users will require running both systems during the transition. These users need to verify the new system works to meet expectations. Most GNU/Linux applications can be run from a proprietary system, either directly or through a remote GNU/Linux server. Often a second PC is not needed for a migration for most users.

Preparation for a full scale installation should include a package server containing all the packages to be installed. Debian supports organizations having mirrors of their web site. The Boot/Install CD-ROM should be pre-configured to point to this server and initial setup. The packages should include selected ones from the GNU/Linux distribution, and local packages and organizational packages.

A clear assignment of responsibility is always required for every organization. The organization must also support the re-assignments of responsibilities. Example: IT is assigned responsibility to all PCs, however there is a small number of unique computers in the engineering department, that neither the IT department nor the engineering department wants IT to manage. So the responsibility is assigned to Engineering. However, basic security requires protection from the unknown engineering systems, so IT gives a special protected engineering network called a “Dead Man Zone” (DMZ). Most routers now can support many Dead Man Zones (DMZs) directly, but a GNU/Linux PC can too.

6 Application development

There are several development environments and data architectures supported in the GNU/Linux. The following sections cover the major concerns.

6.1 Programming languages

The GNU/GCC (GNU's Compiler Collection) is the standard for many platforms including GNU/Linux. The GNU/GCC is the key development component to GNU/Linux and that's why this document uses the term "GNU/Linux". The GNU/GCC includes C, C++, Java, Fortran, Ada, and Objective C.

Ada is recommended for huge or safety critical applications, or embedded test projects. Its rigor and run time checking add a significant level of protection and crash diagnostics.

C++ is the most broadly capable language with good basic protections and great large scale effort support. Major problems missed by a standard C compiler can be caught by the C++ compiler. Costs associated with developing under C++, using its object oriented ability, can quickly pay-back.

Infinite Delta Corp has no experience with GNUstep, using Objective C, but it could be worth investigating.

Java is stricter then C++, but not yet as fully supported, even with Sun Microsystems moving its' Java to fully free. The Infinite Delta Corp does not have any experience with the Java language.

6.2 Development support

There are some Infinite Delta Corp suggested items worth investigating:

1. dselect and dpkg is Debian's package management system.
2. Doxygen is a excellent documentation tool for documenting your code.
3. Octave is a high-level language, primarily intended for numerical computations.
4. GNU gettext tools and the GNU libintl library, allows programs to support multiple languages.
5. libgtkmm is a C++ interface for the popular GUI library gtk+.

Debian's package management is a very powerful administration tool for maintaining packages on a system. Specifically "dpkg –get-selections" is a good way listing packages installed on a system.

GNU/Linux kernel 2.6 has improved the automatic kernel module loading. Explicit GNU/Linux driver support requires an understanding of lsmod and modprobe. Modules can be forced into a kernel at boot from /etc/modules.

6.3 User interface tools

There are two major competing development environments today. First the Gnome GTK and GTKMM which is and has been LGPL. Second, the KDE with Kdevelope is either GPL or Trolltech's Q Public License (QPL) which limits its commercial applications. Recently, Trolltech's Cross-Platform Rich Client Development Framework (QT) by has further opened their system. Again look into licensing issues before committing to a library or a tool.

6.4 Data architectures

A central application server is the quickest and easiest to program, setup, and maintain. However, it has a few drawbacks. The user is dependent on the network to run the application. Data security needs to be very tight on the server, including backup methods.

6.4.1 Robust distributed system

Although initially more expensive, distributed systems are being considered for the following reasons:

1. Better support for people with laptops or who are on the road much of the time.
2. The completed system can be much more robust.
3. A central sever is not required.
4. Computers do not need to be on the network all the time. Only periodic connections are required for a fully operational system.
5. Multiple level security is easier to maintain.
6. Higher performance levels are possible.
7. Lower cost networks can be considered because network down time is less costly.

Caution, some applications claim to use a fully robust distributed architecture, however, they still are dependent on the network to be fully operational. Disconnecting the network is a simple robustness test of any application. Most file server and SQL server based applications fit into the network dependent category.

The mail system is an excellent example of an application that supports distributed, central server, and mixed configurations. The "exim" mail application can be configured as a very simple distributed system with no single server and one in which traffic waits until the network is operational. This type of configuration requires exim placement on each GNU/Linux PC. Each running exim directly gets and sends mail to the other PCs. No central server is required for this configuration.

Alternatively, exim can be configured as a single central mail server. which forces all mail through a single machine.

In a mixed configuration, a exim mail server can be placed on the gateway to allow special protections and filtering for both incoming and outgoing mail. In this case the central server is just for a targeted set of mail; all other mail remains independent of the server and has all the benefits of being distributed.

A robust distributed system requires a distributed system architect who recognizes that requirements and costs drive the architecture. A robust has less dependence on the network, lower costs when the network is down and less stress for IT. A distributed architect should be consulted before committing to a distributed system.

7 Infinite Delta Corp proposals

Infinite Delta Corp is an engineering firm. Infinite Delta Corp's goal is to provide capability optimization yielding the largest gains for most organizations.

Infinite Delta Corp can supply consultants at competitive rates. However, Infinite Delta Corp can also support Xyz LLC by offering a customized Electronic Capabilities Management system as a Debian install package.

7.1 Consultants

Consultants can be a great way to get started down a new path. Their experience often allows great initial progress and avoids costly mistakes. A consultant can train people, write plans, and

review designs. However, unless properly managed or motivated, consultants can manipulate profit for themselves and/or long term personal job security. The best consultants recognize that a job well done has a clean finishing point, while less scrupulous consultants seek to keep their clients permanently and continuously dependent. Infinite Delta Corp seeks shorter term connections which yield maximum customer benefit.

7.2 Electronic Capabilities Management

In addition to consultant services, Infinite Delta Corp offers the Electronic Capabilities Management (ECM) which was developed to help companies monitor and manage their resource utilization. Electronic Capabilities Management is more than a simple distributed resource management tool. Accounting systems and human resource tools that tie into payroll are examples of simple resource management tools.

A Electronic Capabilities Management system differs by managing combinations of resources to create different capabilities. An example: A truck and a driver combined create a delivery service. A plumber and the same truck can deliver and install a new plumbing system. The Electronic Capabilities Management can optimize resources and schedules to make the most out of resources available.

Resources can be anything and everything that can be delivered or used. Generally capabilities include resources such as personnel, equipment, hardware, computers, software, systems of any kind, lab equipment, tools, and any third party resources. Capabilities can also include other capabilities, essentially creating a hierarchy of capabilities.

To build a building is a capability. It is a good example of a hierarchy of capabilities. The building capability is dependent on many sub capabilities such as plumbing, construction, electrical, etc.

More often than not a given capability has associated costs, risks, restrictions, and further dependencies on other capabilities. Costs and schedules are easily derived and optimized when organized as capabilities.

The Electronic Capabilities Management package is new and not yet in the free public domain. It has only a renewable energy demonstration which will be soon on Infinite Delta Corp's web site.

Infinite Delta Corp needs a Request For Proposal (RFP) from Xyz LLC which includes supported languages, initial capabilities and resources to be managed, targeted schedule and budgets. Please include incentives such as publicity with actual data and or statistics available to the public. Xyz LLC needs to select these resources to help support their end customer. It may include employees, students, available services, and equipment. Please include future considerations in the RFP.

Note, any Infinite Delta Corp software contract will have validation and verification paragraphs for completing milestones in the contract. These are explicit tests for acceptance of each milestone. They need to be carefully reviewed by all parties. Infinite Delta Corp could also research requirements of target users. Any initial thoughts or concerns are welcome in the RFP.

Electronic Capabilities Management promotes a very open and secure system. Please include any viewing restrictions in the RFP. Some information such as pay, costs, personal, schedule, etc may need to be restricted to different levels and possibly kept very separate.

A payment to be determined by Xyz LLC to Infinite Delta Corp included with the RFP will determine the seriousness and priority which Infinite Delta Corp addresses the RFP. The payment is only to cover the costs of the proposal. Xyz LLC can use the entire proposal created by Infinite Delta Corp's in any manner they choose, including using it as a template for other software or service providers.

The final Electronic Capabilities Management system will be under GNU's GPL and LGPL licensing.

8 Summary

Infinite Delta Corp hopes this paper helps in Migrating Infrastructure to GNU/Linux and believes this is the correct direction for many organizations. Getting your critical players familiar with GNU/Linux, and embracing the paradigm differences is the key to long term success. Thank you for considering Infinite Delta Corp for your support needs.