Database Creation and Implementation Using Open Source Software

Shelby Patru
Western Michigan University, spatru@gmail.com

Follow this and additional works at: http://scholarworks.wmich.edu/honors_theses

Part of the Computer Sciences Commons

Recommended Citation
Shelby Patru, having been admitted to the Carl and Winifred Lee Honors College in Fall 2002, successfully presented the Lee Honors College Thesis on April 21, 2006.

The title of the paper is:

"Database Creation and Implementation Using Open Source Software"

Dr. Alan Rea, Business Information Systems

Dr. Kuanchin Chen, Business Information Systems

Dr. Muhammad Razi, Business Information Systems
Database Creation and Implementation Using Open Source Software

Shelby Patru

Thesis Chair: Dr. Alan Rea
Thesis Member: Dr. Kuanchin Chen
Thesis Member: Dr. Muhammad Razi

Western Michigan University
Haworth College of Business
Lee Honors College Honors Thesis
Spring 2006
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements</td>
<td>4</td>
</tr>
<tr>
<td>Abstract</td>
<td>5</td>
</tr>
<tr>
<td><strong>Open Source Software Information</strong></td>
<td></td>
</tr>
<tr>
<td>History of Open Source Software</td>
<td>6</td>
</tr>
<tr>
<td><em>Linux Popularity Diagram</em></td>
<td>10</td>
</tr>
<tr>
<td>Pros and Cons of Open Source Software</td>
<td>13</td>
</tr>
<tr>
<td>Open Source vs. Proprietary Software</td>
<td>16</td>
</tr>
<tr>
<td><em>Power Distribution Diagram</em></td>
<td>17</td>
</tr>
<tr>
<td>The Open Source Maturity Model</td>
<td>19</td>
</tr>
<tr>
<td><em>Market Segment Graph</em></td>
<td>22</td>
</tr>
<tr>
<td><em>OSMM Table</em></td>
<td>24</td>
</tr>
<tr>
<td><em>OSMM Recommended Scores Table</em></td>
<td>25</td>
</tr>
<tr>
<td>Apache Information</td>
<td>26</td>
</tr>
<tr>
<td><em>Apache Popularity Diagram</em></td>
<td>27</td>
</tr>
<tr>
<td>MySQL Information</td>
<td>29</td>
</tr>
<tr>
<td><em>MySQL Response Time Diagram</em></td>
<td>30</td>
</tr>
<tr>
<td>PHP Information</td>
<td>32</td>
</tr>
<tr>
<td><em>PHP Usage Diagram</em></td>
<td>33</td>
</tr>
<tr>
<td><em>PHP/Apache Penetration Rate Diagram</em></td>
<td>33</td>
</tr>
<tr>
<td>OpenOffice.org Information</td>
<td>34</td>
</tr>
<tr>
<td><em>Microsoft Word Toolbar/OpenOffice.org Toolbar Comparison</em></td>
<td>35</td>
</tr>
</tbody>
</table>
## GNP Acoustical Database Overview

Business Overview 37
Business Processes 37
Current Means of Organization 38
Purpose of the Database 39
Database Definition 40
Open/Closed Job Status Database Forms 41

*Screenshots of Database Forms* 41-2

OpenOffice.org Connections with MySQL 42
Data Dictionary 43
Normalized Relations 44
Entity Relationship Diagram and Assumptions 45

*Entity Relationship Diagram* 45

Personal Reflections 46
Bibliography 48

Appendix A 51

*Viewing Database Information* 51

Appendix B 52

*Adding Information to the Database* 52

Appendix C 53

*Deleting Information from the Database* 53-4

Appendix D 55

*Searching the Forms* 55-6
Acknowledgments

I would like to express my gratitude to my thesis chair, Dr. Alan Rea, whose expertise and understanding added considerably to my thesis experience. I appreciate his vast knowledge and skill in the area of open source software and advanced computing topics. I would like to thank the other members of my committee, Dr. Kuanchin Chen and Dr. Muhammad Razi for the assistance they provided at all levels of this project. I would like to thank my father for allowing me to use his business in order to create a real world project that allowed me to explore new software and database capabilities.

I would also like to thank my family for their unconditional support they have provided me throughout all my years. Without them, I would not have been able to achieve many of the personal and academic goals that I set for myself. Finally, I would like to thank my boyfriend for his unwavering support in all aspects of my life. Without him, I would not be the person that I am as I now prepare to embark on my professional life.
Abstract

This project is designed to investigate the use of open source software and utilize this software to create a database for GNP Acoustical. This honors thesis will focus on the history of open source software, the pros and cons of using this software, a brief look at open source vs. proprietary software, the open source maturity model and the specifics of major software packages in the open source field that could be used to complete this project. Open source software was chosen as the subject for this project because of its increasing popularity among worldwide corporations.

This project was going to initially be created using the Apache server and PHP scripting language to create a web based database. After reviewing the needs of GNP Acoustical, it was decided that a better route would be to use the MySQL Server and OpenOffice.org. Apache and PHP are still discussed in this paper because of their wide popularity within the open source community.

The database will be created and maintained using MySQL Server and MySQL Administrator and will be accessible to its user (George Patru) via forms created using OpenOffice.org. These software packages will be discussed in later sections. Following the software research, a thorough breakdown of the GNP Acoustical company and the specifics of the database implementation will be discussed.
History of Open Source Software

While open source software (abbreviated as OSS) became a revolution in 1998, its roots can be traced back to 1970. First though, one must understand what OSS really is. “…OSS is software that is licensed to guarantee free access to the programming behind the precompiled binary code, otherwise called the source code” (Bretthauer). This access to the source code allows a user to install the software on any platform without an additional purchase of the software. Unlike freeware and public domain software, OSS is copyrighted and distributed with licensing terms designed to ensure that the source code will always be available to any user (Bretthauer).

Richard Stallman is responsible for laying the groundwork of what would be called the open source movement. In the 1970s, he worked as a programmer at the Artificial Intelligence Lab located at MIT. There he used a locally developed operating system called Incompatible Timesharing System. Within the community that he worked, everyone shared their software meaning that “anybody was welcome to come and take a look, and take away a copy, and do whatever he wanted to do” (Bretthauer). Eventually, the lab replaced this software with a new proprietary operating system.

As Stallman’s free software community dissolved, he decided to create an operating system “…complete with all necessary software tools, such as editors, compilers, and utilities, that should be UNIX-compatible so that programmers could use it without having to learn a new operating system” (Bretthauer). Stallman called this new piece of software GNU (pronounced guh-new) which stood for “GNU’s Not Unix”. By 1985, he released a piece of software entitled GNU Emacs. It was made available free of charge via an anonymous FTP. Since access to the
internet was not very common at this time, he also offered to send the software package to users for a fee of $150.

Since people began questioning this concept of free software and the $150 charge, it is necessary to note Stallman’s definition of “free software”. He explains that “when I speak of free software, I’m referring to freedom, not price” (Bretthauer). Stallman then defined free software as possessing the following essential freedoms:

- You have the freedom to run the program for any purpose.
- You have the freedom to modify the program to suit your needs. (To make this freedom effective in practice, you must have access to the source code, since making changes in a program without having the source code is exceedingly difficult).
- You have the freedom to redistribute copies, either gratis or for a fee.
- You have the freedom to distribute modified versions of the program, so that the community can benefit from your improvements.

As GNU began to gain popularity, Stallman needed a way to ensure the protection of his software from being used in proprietary packages. To guarantee this protection, he developed the concept of copyleft. According to Stallman, copyleft is defined as being a program that is first copyrighted “...then we add distribution terms, which are a legal instrument that gives everyone the rights to use, modify, and redistribute the program’s code or any program derived from it but only if the distribution terms are unchanged” (Bretthauer). With copyleft, the code and the freedom for the code become legally inseparable. Proprietary software developers use copyright to take away the users’ freedoms but copyleft is used as a way to guarantee their freedom.
This method was used to copyleft GNU along with a license agreement called the GNU General Public License (GNU GPL). The first version of this license was released in 1989 and the second (and current) version of this license was released in 1991.

To support the development of GNU, Stallman then founded the Free Software Foundation in October of 1985. This foundation is "a tax exempt charity that raises funds to promote the freedom share and change software" (Bretthauer). While the Free Software Foundation accepts donations, most of its income has always come from the sales of copies of free software and other related services. He believes that free software allows businesses to have control over what their software does or does not do. He feels that when a program is proprietary, users sometimes cannot tell what is going on in the background. In his research, Stallman has found that it is not unusual for proprietary software to “snoop” on a user’s activity and report this information back (Bretthauer).

By 1991, Stallman and his group of programmers had finished everything for GNU except the kernel, the part that ties the entire system together. It was at this time that Linus Torvalds had released the Linux kernel. Torvalds and others combined it with the GNU operation system to make a complete software package. This operating system has since been referred to as Linux.

Before the release of the Linux kernel version 0.02, Torvalds posted the following message to a newsgroup:

“Do you pine for the nice days of Minix-1.1, when men were men and wrote their own device drivers? Are you without a nice project and just dying to cut your teeth on an OS you can try to modify for your own needs? Are you finding it frustrating when
everything works on Minix? No more all-nighters to get a nifty program working? Then this post might just be for you” (Bretthauer).

His excitement for the movement of free software inspired many programmers to give this new operating system a try. After the completion of Linux, Torvalds focused on the technology and the community that built it in the following statement:

“Linux today has millions of users, thousands of developers, and a growing market. It is used in embedded systems; it is used to control robotic devices; it has flown on the space shuttle. I’d like to say that I knew this would happen, that it’s all part of the plan for world domination. But honestly this has all taken me a bit by surprise. I was much more aware of the transition from one Linux user to one hundred Linux uses than the transition from one hundred to one million users” (Bretthauer).

This statement shows the difference between Linux and operation software packages such as GNU, BSD or Apache. Those packages were developed in a carefully, coordinated way by a small, tightly-knit group of people while Linux, was rather casually “hacked on” by huge numbers of volunteers coordinating together via the Internet (Bretthauer).

By late 1993, Linux was competing with many commercial UNIXs and was hosting more software. According to Eric Raymond, a writer of several open source articles, Linux was a project that was conceived about five years after Microsoft began development of Windows NT. Even though thousands of man-hours and millions of dollars were used for the development of Windows NT, Linux was still considered a competitive alternative (Bretthauer).

At the time, internal reports at Microsoft revealed that Linux “…performed better than many MS products on speed and reliability” (Krishnamurthy). In 1997, Infoworld Magazine gave Linux the “Best Operating System” award and hailed the open source community for its
excellent technical support (Krishnamurthy). Nowadays, many well known companies are using Linux as their software of choice. For example, Google is said to have grown to over 100,000 Linux computers spread over a dozen data centers around the world allowing for up to 200 million searches in an average day (Gustafson). As illustrated below in a 2003 survey, Linux has topped the open source software race (Gustafson).

Linux has further developed a business model that may not seem possible: the concept of selling free software and making a profit. By 1994, numerous distributions of Linux were available for less than $30. One of these, Red Hat, has now become a publicly traded company. Robert Young, the CEO of Red Hat, feels that “…you make money in free software exactly the
same way you do it in proprietary software: by building a great product, marketing it with skill and imagination, looking after your customers, and thereby building a brand that stands for quality and customer service” (Bretthauer).

While the development of free operating systems was spectacular in the open source software movement, many other software packages were being developed between the 1980s and 1990s. These include the scripting/programming language PERL, the programming language Python, the web scripting/programming language PHP, the web server program Apache and the databases MySQL, mSQL and PostgreSQL (Bretthauer).

While the software discussed so far has been referred to as “free software”, the term “open source” wasn’t proposed until late 1997. At this time, a group of people were meeting on a regular basis to discuss spreading awareness of the software tools that had been developed outside the proprietary software development model (Bretthauer). This group decided to register the domain name opensource.org which defined the term “open source”, developed OSI certification and created a list of licenses that meet the standards for open source certification.

The basic definition of open source (as described on the website above) is as follows:

- The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources.
- The program must include source code and must allow distribution in source code as well as compiled form.
- The license must allow modifications and derived works and must allow them to be distributed under the same terms as the license of the original software.
The license may restrict source code from being distributed in modified form only if the license allows the distribution of patch files with the source code for the purpose of modifying the program at the build time.

- The license must not discriminate against any person or group of persons.
- The license must not restrict anyone from making use of the program in a specific field of endeavor.
- The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
- The license must not be specific to a product.
- The license must not contaminate other software by placing restrictions on any software distributed along with the licensed software (Brethauer).

Overall, the Open Source Definition allows for greater liberties with licensing than the GPL does. In particular, the Open Source Definition allows users greater freedoms when mixing proprietary and open source software (Brethauer).

The licenses listed above differ on four key dimensions:

1. “whether they allow the OSS program to be mixed with the non-free software;
2. whether modifications can be made private and not returned to the author(s);
3. whether the program can be relicensed by anyone; and
4. whether it contains special privileges for the original copyright holder” (Krishnamurthy).

The leading licenses for OSS can be seen in the table below along with the aforementioned dimensions.
Comparison of open source licenses

<table>
<thead>
<tr>
<th>License</th>
<th>Can be mixed with non-free software</th>
<th>Modifications can be made privately and not returned to author</th>
<th>Can be relicensed by anyone</th>
<th>Contains special privileges for the original copyright holder over user's modifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Public License (GPL)</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>GNU Library General Purpose License</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Berkeley System Distribution</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Netscape Public License</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Mozilla Public License</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Public Domain</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

(Source: www.perens.com/Articles/OSD.html)

The Open Source Development Lab has also emerged as a non-profit organization dedicated to advancing open source software. They have launched a formal initiative to get Linux onto the corporate desktop. So far, HP is only PC maker to offer Linux PCs with OpenOffice (an open source alternative to Microsoft Office), which it began marketing in Asia in March 2004 (Gustafson). For now, only time will tell whether or not the corporate world is ready to give up their Microsoft heavy systems in order to switch to an open source system.

Pros and Cons of Open Source Software

If a company as popular as Google is using the open source operating system Linux to run their search engine, there must be advantages to the OSS products that are making them appealing to larger companies. This section will investigate these advantages along with the disadvantages associated with the use of OSS. This section is designed to give the facts on both sides while maintaining an unbiased opinion.

Many OSS products are free, though some, as explained in the above section, do come with a small fee. Because of the cheaper cost of the software, “this lowers the risk of trying the product and improves the chances of rapid diffusion among the appropriate target audience” (Krishnamurthy). Of course, the costs must be examined comprehensively on a case-by-case
basis because costs may decrease in one area and increase in others. What an organization saves in license fees may be offset by increased technical support, training and other costs outside the bounds of the organization’s IT department (Gustafson).

Robert Fanini, CEO of GroundWork, a commercial firm that offers a suite of IT management solutions using open source software had compiled a list of benefits that he has found appeal to IT executives. “Among them:

- Elimination of licensing fees,
- Lowering deployment costs – because you only install what you need,
- Cutting system administrative overhead costs – which arise because IT shops only manage what they need or what they’ve installed,
- Lower hardware costs – thanks to inexpensive Linux boxes, rather than higher-end proprietary Unix systems” (McCarthy).

Using the open source method of software creation, there is a potentially endless number of developers and testers that can work on the package. There is also the belief that the development process may be faster and smoother because developers can use available and pre-tested components rather than having to create software from scratch (Gustafson). “In the words of Linus Torvalds:

There are lots of advantages in a free system, the obvious one being that it allows more developers to work on it, and extend it. However, even more important than that is the fact that in one fell swoop it also gave me a lot of people who used it and thus both tested it for bugs and tested it for usability. The ‘usability’ part comes from the fact that a single person (or even a group of persons sharing some technical goal) doesn’t even think
of all the uses a large user community would have for a general-purpose system”
(Krishnamurthy).

One of the major advantages of OSS is the freedom to mix and match software components so that a user can get exactly what they want. According to Torvalds, one of the main ideas with free software is not so much the price and not having to pay cash for it, but the fact that, with free software, you are not tied to any one commercial vendor. You might use some commercial software on top of Linux, but you are not forced to do that or even to run the standard Linux kernel at all if you do not want to (Krishnamurthy).

While with traditional proprietary software a user will need to wait on hold for customer assistance, open source has a different solution. Because of the global network of users and testers, anyone using OSS has an engaged community willing to answer questions. This 24 hour a day availability is a feature that appeals to both developers and users alike.

Some investigators believe that since the open source structure leads to more peer review, this must in turn lead to a more reliable software package. They believe that having the global community be able to find and fix its own bugs will create a much more dependable product in the long run. The issues of security can also be raised under this point. Many commercial and governmental agencies are worried that open source developers are too much ‘hacker’ and too little ‘engineer’, combining together solutions without going through a structured software engineering process like most proprietary software vendors (Vieg).

Some would argue that if there is strong peer review, security flaws will likely be caught. The better the structure of the peer review, the better the security (Krishnamurthy). Due to the overwhelming number of testers and developers, problems can be caught early and fixes can be prepared quickly. In some cases, “…monetary rewards are offered to further encourage
developers to find bugs. After several high profile flaws were found in browsers, the Mozilla Foundation began offering $500 for every serious bug found by security researchers in its open source software” (Gustafson). As stated by Whitfield Diffie, chief security officer of Sun Microsystems, “it’s simply unrealistic to depend on secrecy for security in computer software. You may be able to keep the exact workings of the program out of general circulation, but can you prevent the code from being reverse-engineered? Probably not. The secret to strong security: less reliance on secrets” (Gustafson).

One criticism to OSS is the belief that it appeals to mostly high end technical users. This could explain why a product like Linux has experienced positive feedback from the server side but disappointment from the client side. “The mainstream user wants performance and features over reliability, one easy-to-use suite rather than a choice, and probably does not care about access to the source code” (Krishnamurthy). Because of this, many developers are working to make the Linux interface appear more like Windows.

Open source communities lack the resources to market their products through traditional media (Krishnamurthy). As a result, information about new software is mainly transferred by word of mouth. While this might gain awareness from developers and technically savvy users, the mainstream public is not aware of the new software.

Open Source vs. Proprietary Software

After examining the above advantages and disadvantages of OSS, it is important to compare this software to the more commonly used proprietary software. In this paper, the term proprietary refers mostly to Windows based software as Microsoft seems to be the most commonly used among businesses.
"In the world of software, access to the source code is a prerequisite for innovation" (Krishnamurthy). Without this access, product innovation is restricted only to the author of the software. By providing users of OSS with the source code, they become empowered to change the software as they see fit. This distribution of power can be seen in the figure below.

When deciding whether or not to use proprietary or OSS, the article entitled *Proprietary Source vs. Open Source* created a table outlining the major differences between the two. This table can be seen below.
From this table, it is obvious that the licensing and access opportunities are much looser for OSS then the proprietary software. It is also important to note that the technical skills needed for proprietary software are lower than that of OSS. This could be due to the fact that most users are more familiar with current proprietary software and are not aware of the OSS options currently available. While proprietary software does offer formal technical support, there is nothing to indicate that it is any more helpful than the many newsgroups available to help with an open source problem. In fact, this helpfulness is one of the highlights of the open source movement, which is demonstrated above under the size of the development team criteria.

As it was stated in the above section, security is again an issue when discussing OSS and proprietary software. Some believe that “simply put, you get what you pay for. Commercial development organizations have a powerful motivation to get security right: Their livelihoods depend on it” (Allison). Commercial firms employ people who are dedicated, hardworking and have a passion for security. These people get paid to do the tedious work of testing for security.
flaws, something that is not especially interesting to open source volunteers. “Security flaws are not like ordinary bugs. They are subtle and complex, and often require specialized or in-depth knowledge to identify” (Payne).

As for OSS, most packages are written by people for whom programming is not a chore. It is a craft, and they take great pride in doing their work properly. Away from the demands of marketing and management, they are able to create the code that they want to write, not the code that will make the most money (Allison). There is also the concept of the global community who is supposedly working just as hard as a proprietary company to work out any bugs in the code.

It is hard to argue with the statistics as well. In 2000, “…Red Hat issued 137 security bulletins for Linux alone, while Microsoft issued 100 bulletins covering its full range of products” (Allison). On the other hand, “A Michigan insurance company has raised hacker insurance premiums for sites running Microsoft Windows NT to up to 15 percent more than those for sites running Linux. Insurance companies don’t deal in opinions – they deal in facts” (Allison). Based on these figures, the topic of security still seems to be something of a mystery. Since the internet will never be seen as 100 percent safe, it is hard to conclude which kind of software will offer the most security.

The Open Source Maturity Model (OSMM)

Most open source software is nowhere near as complete as it would be if it was being produced by a big name company such as Microsoft. Because of this reason, organizations wanting to convert their programs to open source must have a way to measure the open source software’s maturity. This means of measurement is called the Open Source Maturity Model (OSMM). The overall purpose of the OSMM is “…to make a quick assessment of the maturity level of a given open source product” (Golden 81).
The first challenge of determining software’s maturity level is to determine which user will be implementing it. Geoffrey Moore, the author of *Crossing the Chasm*, has identified three distinctive groups of PC users. These groups are as follows: early adopters, pragmatists and late adopters. For the purposes of the OSMM, only the early adopters and pragmatists will be discussed. “Early adopters are comfortable using “unfinished” products, whereas pragmatists prefer to wait for the “whole” – or mature – product” (Golden 62).

Early adopters believe that innovative technology enables them to develop new product or service offerings that will dramatically reduce their cost structure (Golden 63). The goal of an early adopter is to apply new technology in order to gain a competitive advantage. However, these new technologies do come with a price. New software often lack complete functionality, might have quality issues, and require users to experiment causing frustration among many (Golden 65).

Wal-Mart is a real world example of an early adopter. Wal-Mart has managed to successfully restructure the retail industry using its new application of supply chain technology. They have an innovative way of using inventory tracking, supplier cost assessment, and business intelligence that has allowed them to continue offering their goods at a low price (Golden, 63-4). Nowadays, Wal-Mart is venturing even further into new technology by researching Radio Frequency Identification chips which would enable the company to have a more efficient way of tracking inventory. Another example of an early adopter is Amazon. The book industry was completely restructured when Amazon began offering the convenience of online shopping along with competitively low prices. Not only were they a vendor for books, but they supplied many other things that consumers could not get from a regular bookstore such as “…book recommendations based on previous selections, buying circles… and the ability to speed
transactions by storing user details” (Golden 64). Along with implementing this new way of buying books, Amazon has also reduced the company’s IT costs by using mostly open source software to run the company (they run their entire computing infrastructure using Linux).

Pragmatists, on the other hand, seek efficiency and cost effectiveness. “They can be characterized as “fast followers” who begin implementing a technology once it is successfully applied within the industry” (Golden 65). They prefer to have a product that is easy to understand, fully documented, fully supported and supports vast training of all employees. The goal of a pragmatist is to determine how new technology will integrate with their existing business strategy while causing the least amount of problems (Golden 67).

Target retail stores are an example of a pragmatist. Target responded to Wal-Mart’s low prices by creating its own supply chain strategy. “Target offers everyday low pricing on its goods, but also offers promotions on specific goods to drive traffic” (Golden 66). Unlike Wal-Mart, Target has chosen to focus on more specific goods in order to support its own business strategy. Another example of a pragmatist is Barnes and Noble (B&N). Responding to the online shopping wave, B&N decided to implement a strategy that would enable them to sell books both in stores as well as online. Though B&N only offers online shoppers books, music and videos they do allow online purchases to be conveniently returned in-store. Their website also allows a user to “…search the inventory of local stores for a specific book, which allows users to see if a book is available locally for immediate purchase” (Golden 67).

As seen in the chart below, pragmatists are the dominant users within the IT market.
In fact, early adopters make up only 15 percent of the market segment (Golden 69). For a technology company to be successful, it is more important to address the needs of the pragmatists. If a pragmatist wishes to partake in the open source movement, they will need to become much more active and create their own mature product that will meet all their requirements.

Before discussing the way of determining a product’s maturity, mature must be defined according to certain software characteristics. These characteristics are outlined below (Golden 71-2):

- A full feature set
- High quality
- Longevity in the IT marketplace
- Easy administration
- Good customer support
- Mature behavior in error situations

Maturity is used to determine how well a product is suited for the particular use it is intended for. Because of the nature of the open source community, IT organizations cannot expect the same
features that they might get from commercial software. “Open source offers IT organizations much more control of their destiny; it also imposes much more responsibility for their product choices” (Golden 74).

With more than 80,000 open source products available, it is important for an organization to have a benchmark of the maturity they require in their software. This is where the OSMM comes in. The OSMM assesses a products maturity in the following three phases (Golden 75):

- Assess on the product’s maturity and give a maturity score.
- Define a weighting scale for each element based on the requirements of each specific organization.
- Calculate the product’s overall maturity score.

The first phase of the OSMM is to identify the key elements of the software and the maturity level of each element. For this example, the elements are as follows: product software, support, documentation, training, product integration and professional services. Each element chosen by an organization is then assigned a maturity score based on the following four-step process (Golden 76):

1. Define organizational requirements
2. Locate resources
3. Assess element maturity
4. Assign element maturity score on a scale of 1 – 10

By assigning a score to each element maturity, the organization can clearly see which requirements are more valued and this can make the decision of which software to use that much easier.
The second phase of the OSMM is to assign a weighting factor to each element chosen in the previous phase. The weight given to each element reflects the level of importance that the organization places on it. The only requirement for the element weighting is that the scores of all the elements must total 10.

The third phase of the OSMM is to calculate the product’s overall maturity score. All the element scores are summed to give an overall product maturity score from 1 – 100. From here, an organization can decide if the maturity score is at a satisfactory level to implement the software into their infrastructure. An example of a complete OSMM table is pictured below (Golden 77).

<table>
<thead>
<tr>
<th>Element</th>
<th>Potential Score</th>
<th>Actual Score</th>
<th>Weighting Factor</th>
<th>Element Weighted Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>Support</td>
<td>10</td>
<td>8</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>Documentation</td>
<td>10</td>
<td>6</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Training</td>
<td>10</td>
<td>8</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Integration</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>Professional Services</td>
<td>10</td>
<td>6</td>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>

| Total Product Maturity Score | 78 |

The total product maturity score should be assessed according to the requirements of the organization. Recommended scores vary depending on whether the organization considers itself to be an early adopter or a pragmatist. Scoring is based on how the company categorizes itself along with three different product environments: experimentation, pilot and production.

The experimentation environment allows an organization to “...begin working with a new technology to better understand how it works and what uses might be made of it” (Golden 86). Because more immature products can be used for experimentation, the maturity scores tend to be low. The pilot environment “…seek[s] to replicate the conditions that a product will operate in under production loads, but they are primarily oriented toward learning how
everything about the product works and to ensure that all elements of the product exist and are operational” (Golden 86). Because the pilot environment is replicating the actual work environment, the maturity scores tend to be higher. Finally, the production environment requires that the product be working correctly in all aspects (Golden 87). This includes full functionality of the software, full training of the staff, defined operational procedures and a support team ready if things should go wrong. This environment requires the highest maturity scores. The recommended minimum scores can be seen in the table below for each user.

<table>
<thead>
<tr>
<th>Purpose of Use</th>
<th>Type of User</th>
<th>Early Adopter</th>
<th>Pragmatist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experimentation</td>
<td>25</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Pilot</td>
<td>40</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>Production</td>
<td>60</td>
<td>70</td>
<td></td>
</tr>
</tbody>
</table>

It should be noted however, that these scores are only recommendations. Even though a product might not meet the maturity scores mentioned above does not mean that it will not still suit the organization. The purpose of the recommendations is to provide a good starting point for determining how mature a product needs to be for a given purpose.

The OSMM is designed to enable a small group from an organization to spend no more than three to five days developing a maturity score for a product. Since no model can completely reflect an organization’s requirements, the OSMM is designed to “…assess whether a product is good enough for a thorough hands-on assessment, including a test installation and pilot exercise” (Golden 82). While there is no substitute for a detailed technical assessment of a product, the OSMM allows companies to more easily determine if open source software is right for them.
Apache Information

The Apache HTTP Server Project (just shortened to Apache for this report) was first released in the spring of 1995. It is “…a collaborative software development effort aimed at creating a robust, commercial-grade, featureful, and freely-available source code implementation of an HTTP (Web) server” (Apache Web Site).

In the winter of 1995, Rob McCool had developed the most popular Web server software, called the public domain HTTP daemon. However, once McCool left the National Center for Supercomputing Applications, the construction of this project began to slow. Thanks to the help of a small group of developers, they worked together to create bug fixes and enhancements so that Apache version 0.6.2 would be possible.

Over the next year, more programmers came together to help implement new features and design new server architecture. By August of 1995, version 0.8.8 was available for download. “After extensive beta testing, many ports to obscure platforms, a new set of documentation (by David Robinson), and the addition of many features in the form of our standard modules, Apache 1.0 was released on December 1, 1995” (Apache Web Site). Since its release, it has consistently held the top spot for most popular server. This can be seen in the Netcraft web survey graph pictured below.
Based on the CNET news website, in January of 2004 “Apache’s market grew from about 62% to about 67%, while that of [Microsoft] IIS dropped from 27% to 21%”. Mike Prettejohn, the director of Netcraft, believes Apache has done well, because of its stability, good security record, straightforward administration and free cost (Broersma).

This fantastic security record is also another factor earning raves for Apache. “A study of Apache security advisories dating back to Apache 1.0 shows the server’s last serious problem… was announced in January 1997” (Dyck). Since then, there have only been a handful of problems, all of which were fixed quickly. The reason for this reliable security is, according to Dyck, due to three key reasons.

First, the installation of Apache does not install a lot of extras to the machine. In fact, all you get is a bare-bones Web server. Compare this to Microsoft’s IIS, which is installed along with seven library file extensions plus FrontPage Server extensions. Second, “all Apache components run as nonprivileged users, so if a buffer overflow [does] occur damage is limited”
(Dyck). Finally, the installation of Apache puts all the configuration settings into one file (httpd.conf) which provides a single point of configuration. With IIS, settings can be changed in several different locations.

In 1999, the Apache Software Foundation was formed to “…provide organizational, legal, and financial support for the Apache HTTP Server. The foundation has placed the software on a solid footing for future development, and greatly expanded the number of Open Source software projects, which fall under this Foundation's umbrella” (Apache Web Site). This foundation includes a global community dedicated to developing the server, contributing ideas and documenting problems or solutions.

The Apache group has prided itself on keeping this powerful server as an open source tool. As seen on their web site, “We believe that the tools of online publishing should be in the hands of everyone, and that software companies should make their money by providing value-added services such as specialized modules and support, amongst other things”. It is the belief of the Apache Foundation that allowing the software to be free leads to more feature enhancements, bug fixes and overall user support. It is the argument of this foundation that their strength comes from the fact that the software is free, and if it were made “not free” then Apache would suffer tremendously, even if that money were spent on a real development team (Apache Web Site).
MySQL Information

MySQL was developed in 1995 in Scandinavia by David Axmark, Allan Larsson and Michael "Monty" Widenius and is now “...described as the world’s most popular open source database” (Langley). Since its creation, it has fast become part of the LAMP (Linux, Apache, MySQL, Postgres/Perl/PHP/Python) platform.

Currently, MYSQL is available both free under an open source license and with a commercial license shipped with other products such as Novell’s Netware. Likewise, support is available through the MySQL community, or through a commercial subscription (Langley). So far, this software package has obtained a six million plus user base based on its performance and price.

With so many other database systems to choose from, namely Oracle which is widely used by corporations today, what is it that makes MySQL so special? It is said to offer “...minimalist but functionally rich features, yielding speed and compactness. Other claims include database licensing costs reduced by 90%, system downtime reduced by 60%, and administration, engineering and support costs cut by up to 50%” (Langley). Because of these benefits, MySQL has large corporate users including AOL and Google, Dow Jones, Associated Press, Nasa, Suzuki, Alcatel, Ericsson and Siemens (Langley).

A survey conducted by eweek.com testing various databases on multiple categories yielded the result below.
This figure is based on the time it took each database “…to complete six bookstore user action sequences, weighted by each sequence” (Dyck). It is shown here that MySQL and Oracle9i offered the fastest response times in this category. Overall, this survey found that MySQL and Oracle9i outperformed every database system and had the best performance and scalability in every category (Dyck).

Evan’s Data Corporation released survey results in the fall of 2005 indicating that the deployment of MySQL has increased by more than 25 percent. This company also found that “…proprietary database servers are more likely to have suffered a security breach in the past year than their open-source brethren. Only about 9 percent of developers working with open-source databases suffered breaches [in 2004], compared with about 15 percent of those working with proprietary databases”.

MySQL AB, the company that markets MySQL has the following 10 benefits listed on their website.
1. Scalability and flexibility.
3. High availability.
4. Robust transactional support.
5. Web and data warehouse strengths.
7. Comprehensive application development.
8. Management ease.
9. Open source freedom and 24x7 support.
10. Lowest total cost of ownership.

Along with MySQL, this project will utilize one of the server’s add-on features. This feature, entitled MySQL Administrator, is a “…powerful visual administration console that enables you to easily administer your MySQL environment and gain significantly better visibility into how your databases are operating” (MySQL Web Site). In the past, all database queries and other functions had to be performed via the command line, but this tool allows the user to perform the desired operations through a graphical interface. This includes the ability to configure servers more easily, administer new user information and monitor the health of all databases.

Because of the easiness of this feature, there are many important advantages to its use. Now, an administrator can “…rapidly create consistent backups and restore data by simply visually selecting the schemas and tables [they] want to back/restore from [their] hierarchical data directory” (MySQL Web Site). Server information is also easily accessible and this interface provides visibility of server status, version number(s), IP address and connection
information. Server problems can also be diagnosed quicker using the MySQL log files which collect error and binary logs. Perhaps the most important feature of any open source software is the availability. MySQL Administrator is available for Microsoft Windows 2000 to Server 2003, Linux and Mac OS X to 10.4.

**PHP Information**

PHP is a server-side, cross-platform, HTML embedded scripting language (Cullen). This software package is essentially used as a “programming language and script-parsing engine that work in unison to provide a way for Web developers to quickly and easily develop dynamic content” (Cullen).

PHP was originally written in the Perl coding language and was only intended for the use of its creator, Rasmus Lerdorf. Over time though, more developers began to come across this software and started asking for different features. “The name of this first package was Personal Home Page Tools, which later became Personal Home Page Construction Kit” (PHP Web Site). Nowadays, PHP stands for PHP Hypertext Preprocessor.

One of the many advantages of using the PHP software is its ability to be run on the server. This means “there is no problem with JavaScript version and browser incompatibilities or users who have turned JavaScript off” (Cullen). PHP’s popularity stems from its ability to easily create dynamic web pages. Dynamic content refers to the ability to take information out of a database and insert it into a web page (Cullen).

PHP’s popularity has grown significantly in the last 5 years. This increase can be seen in the graph below.
Based on this survey conducted by Netcraft, PHP has seen an increase of over 20 million in the past half decade.

Based on the Apache Module Report of PHP, it can be seen that this software currently penetrates about 40% of the market. This statistic can be seen in the figure below.

PHP’s popularity among users can be attributed to its capacity to work on many different platforms. “While PHP is closely identified with Apache, it will work with other Web servers,
including Microsoft’s IIS, Netscape, iPlanet, and Oreilly Website Pro. PHP will also run on Windows and Macintosh operating systems, as well as most flavors of Unix and Linux” (Cullen). PHP can also interface with many types of databases including Oracle, PostgreSQL, MySQL, mSQL, and dBase. It can also interface with databases that support the ODBC standard (Cullen). With this range of connectivity, it is no wonder that PHP has exploded as it has in the open source community.

OpenOffice.org Information

OpenOffice.org (further abbreviated as OOo for this report) was created in Germany from a software package called StarOffice Suite. This software was created in the mid 1980’s and was soon used as the backing for OOo. Since its induction into the open source community, OOo has embraced this environment with open arms. On the OOo web site, they state “OpenOffice.org is an Open Source project. This means, first, that we offer not only a product but a process, and second, that we depend upon the contributions of developers and end users to make that process happen”. Perhaps the most monumental moment for the open source community was when OOo was named as the Open Source Product of the Year for 2006 (DeWolf).

OOo is most simply a multi-platform office productivity suite. The software package contains key applications such as a word processor (Writer), spreadsheet capabilities (Calc), presentation creator (Impress), drawing program (Draw) and database creation software (Base). The user interface was created to look similar to other popular office suites available today. One of the best features of this package is that it “…also works transparently with a variety of file formats, including those of Microsoft Office [and Corel WordPerfect], and the vendor-neutral
OpenDocument standard” (OOo Web Site). The software package is available in over 65 languages and runs on many platforms, from Linux to Windows to Mac OS X.

The OpenDocument standard was developed by OASIS (Organization for the Advancement of Structured Information Standards), and is the “...international body for the development and ratification of e-business standards” (OOo Web Site). The OpenDocument format can be used by any office application without fear of vendor lock-in or licensing terms/fees while allowing the user confidence that their documents can be viewed, printed and edited for generations. In 2005, “Microsoft has said it will not support OpenDocument… [and] has stated that the format doesn’t sufficiently encapsulate the functionality of its Office products” (Brooks). In 2006, the government of Massachusetts made the decision to standardize their office documents according to the OpenDocument standard (DeWolf).

The newest version of OOo (2.0) allows for even greater user familiarity. For example, below is the toolbar for Microsoft Office (top) and the toolbar for the OOo Writer software.

To most users familiar with the MS Word interface, OOo has done an excellent job of creating icons so similar that users may be able to switch with little frustration. Multi-pane views, floating toolbars and desktop integration tools are helping to provide users with a less stressful approach from switching to common proprietary office suites. The menus, toolbars and shortcuts can also be customized to each user’s personal preferences. The advanced user will
also find such tools as “...mail merge, the ability to read and write Microsoft office documents, macro recorders, and Macromedia flash integration” (DeWolf).

While OOo sounds like an easy switch, it may not be this way for an entire corporation. Studies have shown that Microsoft Excel is a hard software application to match and that while OOo’s Calc tries its best there may still be some frustration for users. Organizations are advised to consider how often MS Excel is used before deciding if the open source alternative is right for them. Training costs and reduced productivity due to readjustments to new software should also be considered. “With that said, it’s worth noting that even new Microsoft Office releases require a breaking-in period” so a change to open source may not be a bad idea at all (Brooks).
Business Overview

GNP Acoustical Inc. was established as a Michigan Corporation on June 19, 1986 by George Patru. As an employee of many different construction companies, Mr. Patru felt that he had no job security. Therefore, he started his own business so that he could control his own destiny. The first thing he did to acquire work was he became a member of the Construction Association of Michigan. Through this association, he made many contacts that have been developed through the years. GNP Acoustical provides the installation of acoustical ceilings, mainly in commercial buildings. These ceilings are installed in retail shopping centers, drug stores, medical buildings, office buildings, government buildings, schools and churches.

Business Processes

- Invoice Processes
  
  o Mr. Patru receives a job blueprint from a contractor via mail.

  o After reviewing the blueprint, Mr. Patru creates an estimate for the job based on material costs, size of job and location.

  o Estimate is documented as an invoice and sent to the contractor via fax or mail.

  o If contractor accepts the estimate, a confirmation contract is sent to Mr. Patru and a price is agreed upon by both parties.

  o A new invoice is then drafted to the contractor confirming this price.

  o If a job is set to begin at a later date in the year, this job is filed away into a color coded folder and remains on Mr. Patru’s desk until it is completed.

  o When job is complete, the date of completion and invoice number is recorded by Mr. Patru in a Microsoft Excel document.
The completed job folder is then filed away under the specific contractor name in a filing cabinet.

**Accounting Processes**

- Final cost of each job is recorded into a separate Accounts Receivable Microsoft Excel document.
- At the end of the year, total of accounts receivable column is calculated.
- This total is used to determine total income for income tax purposes.

**Current Means of Organization**

Since Mr. Patru is the president and sole employee of this company, his means of organizing his information is quite simple. For all of his business contacts (suppliers and contractors), he keeps their business cards with their up-to-date information in a binder. Financial information such as past job costs and money spent on supplies and invoices is kept in Microsoft Excel files separated by year. Current and past job information is maintained in file folders stored in a filing cabinet. All current jobs are kept in color coded folders that are stored on Mr. Patru’s desk.

Labor costs for each job are calculated on a per job basis. Mr. Patru’s costs are based on the size and intensity of a job along with the specific location. These costs were calculated based on other companies in the industry and are displayed on a bulletin board in Mr. Patru’s office.

Finally, supplies for each job are decided upon after blueprints have been approved by Mr. Patru. Once the final cost of the project is agreed upon via an invoice, various suppliers are used for specific materials. The most commonly used materials are displayed on a bulletin board. Many materials though must be custom ordered through various construction catalogs or
through the internet. Material costs can vary due to the job size. Because of this, some suppliers provide different prices based on the larger or smaller job.

**Purpose of the Database**

Although the organization for this company is straightforward, I felt that it would give me a good starting ground on creating and maintaining a database on my own. Since I feel that this is the area of the IT industry that I want to pursue, I thought it would be beneficial to get my feet wet doing a database that had to do with an actual company. GNP Acoustical was the most accessible company for me to research and therefore was chosen as the company for my thesis.

I designed this database in hopes of providing Mr. Patru with an easier way of viewing all of his information. The database is designed to be a snapshot of his business at specific periods in time. It is not designed to provide budgeting or job estimations. Because of the complexity of specific jobs, it is easier for Mr. Patru to estimate costs by using various blueprints and supplier catalogs. This database was also created to allow for some flexibility in updating contact information for both contractors and suppliers and for updating current job status along with closing jobs that have been completed.

In terms of the business processes, this database was designed to allow for easier access to all business information instead of using multiple spreadsheets to view various aspects. The accounting processes are calculated using a Microsoft Excel spreadsheet and will remain this way as this is Mr. Patru’s most comfortable way of keeping track of his company’s finances.
**Database Definition**

Four tables were created in designing this database. These tables are as follows: Materials, Contacts, Suppliers and Current Jobs. The database will be created and maintained using the MySQL server along with the add-on MySQL Administrator. The front end interface will be created using OpenOffice.org and will be accessible and updateable to Mr. Patru via forms.

The materials table will contain the following information: a unique material ID number, the name of the material, the material’s product number obtained from either the supplier or common materials used from catalogs, the number of pieces obtained in a single order of the product, the cost of one carton of this material and the cost per piece of the material if it is only sold individually. Materials are ordered for specific jobs and therefore can only be used for one job at a time.

The contacts table will contain the following information: a unique company ID number, the company name, the name of the contact person for the company, a company phone number, a fax number, the email address of the contact person, the address and city from where the company is located. At times, GNP Acoustical is tackling multiple projects so a contact can be involved in more than one job.

The suppliers table will contain the following information: a unique supplier ID number, the company name, the name of the contact person for the company, a company phone number, a fax number, the email address of the contact person, the address and city from where the supplier is located. Each job uses only one supplier but multiple suppliers are kept on hand by GNP Acoustical in case custom materials are needed.
The current jobs table will contain the following information: a current job ID number, the status of the job (whether it is in progress or completed), the invoice number of the job (only applicable if the job has been completed), the cost of completing the job, the start date, the estimated completion date (or completed date if the job is done) and the job name. Since this table will contain information on both open and closed jobs, the jobs will be searched by the terms “open” and “closed” based on the status attribute.

**Open/Closed Job Status Database Forms**

As mentioned above, the job status of each job is determined by the status attribute. This attribute was queried according to the terms “open” and “closed”. Based on the query results, two forms were generated. Since Company ID is a foreign key on the current jobs table, company information is also shown on the job forms. Below are the forms detailing jobs that are closed, followed by the jobs that are currently open.
OpenOffice.org Connections with MySQL

In order to connect the MySQL server with OpenOffice.org a JDBC (Java Database Connectivity) driver had to be installed. The newest version of the driver could be obtained from the MySQL web site. The driver then had to be downloaded and installed into the database that needed to be connected to the server. Correct installation of this server allows the user to see all the tables defined within the MySQL server. From this point, the user can now open these tables and input all the necessary data for the database.
### Data Dictionary

#### CURRENT JOBS

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Data Type</th>
<th>Data Format</th>
<th>Sample Value</th>
<th>PK?</th>
<th>FK?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Job ID</td>
<td>Job ID number</td>
<td>Numeric</td>
<td>Not null</td>
<td>0001</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Status</td>
<td>Open or Closed</td>
<td>Text</td>
<td>Null</td>
<td>Open</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Invoice Number</td>
<td>Invoice Number</td>
<td>Text</td>
<td>Null</td>
<td>01-01</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Cost</td>
<td>Cost of job</td>
<td>Currency</td>
<td>Null</td>
<td>$10,000</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Start Date</td>
<td>Start of job</td>
<td>Date/Time</td>
<td>Null</td>
<td>1/1/2006</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Est. Completion Date</td>
<td>Estimated completion date of job</td>
<td>Date/Time</td>
<td>Null</td>
<td>2/1/2006</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Company ID</td>
<td>Company requesting the job</td>
<td>Numeric</td>
<td>Not null</td>
<td>0003</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Material ID</td>
<td>Material used for job</td>
<td>Numeric</td>
<td>Not null</td>
<td>0006</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

#### CONTACTS

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Data Type</th>
<th>Data Format</th>
<th>Sample Value</th>
<th>PK?</th>
<th>FK?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company ID</td>
<td>Company ID number</td>
<td>Numeric</td>
<td>Not null</td>
<td>0007</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Company Name</td>
<td>Name of company</td>
<td>Text</td>
<td>Null</td>
<td>Sterling Construction</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Contact Name</td>
<td>Name of company contact</td>
<td>Text</td>
<td>Null</td>
<td>Mark Andrews</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone number</td>
<td>Text</td>
<td>Null</td>
<td>248-689-1214</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Fax</td>
<td>Fax number</td>
<td>Text</td>
<td>Null</td>
<td>248-689-4265</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Email</td>
<td>Email of company</td>
<td>Text</td>
<td>Null</td>
<td><a href="mailto:S2patru@wmich.edu">S2patru@wmich.edu</a></td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Address</td>
<td>Company address</td>
<td>Text</td>
<td>Null</td>
<td>67 Miracle Drive</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>City</td>
<td>Company city</td>
<td>Text</td>
<td>Null</td>
<td>Troy</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

#### SUPPLIERS

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Data Type</th>
<th>Data Format</th>
<th>Sample Value</th>
<th>PK?</th>
<th>FK?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supplier ID</td>
<td>Supplier ID number</td>
<td>Numeric</td>
<td>Not null</td>
<td>0008</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Company Name</td>
<td>Name of company</td>
<td>Text</td>
<td>Null</td>
<td>ABC Construction</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Contact Name</td>
<td>Name of company contact</td>
<td>Text</td>
<td>Null</td>
<td>Bob Smith</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone number</td>
<td>Text</td>
<td>Null</td>
<td>810-650-5982</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Fax</td>
<td>Fax number</td>
<td>Text</td>
<td>Null</td>
<td>810-650-9743</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Email</td>
<td>Email of company</td>
<td>Text</td>
<td>Null</td>
<td><a href="mailto:Angelz930@gmail.com">Angelz930@gmail.com</a></td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Address</td>
<td>Company address</td>
<td>Text</td>
<td>Null</td>
<td>1606 Englewood</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Attribute</td>
<td>Description</td>
<td>Data Type</td>
<td>Data Format</td>
<td>Sample Value</td>
<td>PK?</td>
<td>FK?</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------</td>
<td>-----------</td>
<td>-------------</td>
<td>--------------</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>Material ID</td>
<td>Material ID number</td>
<td>Numeric</td>
<td>Not null</td>
<td>0009</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Name</td>
<td>Name of material</td>
<td>Text</td>
<td>Null</td>
<td>White tile</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Product Number</td>
<td>Product number from supplier</td>
<td>Text</td>
<td>Null</td>
<td>M7</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Pieces</td>
<td>Pieces in a carton</td>
<td>Numeric</td>
<td>Null</td>
<td>25</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Carton Cost</td>
<td>Cost of a carton</td>
<td>Currency</td>
<td>Null</td>
<td>$100</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Cost per Piece</td>
<td>Cost per piece in a carton</td>
<td>Currency</td>
<td>Null</td>
<td>$2.50</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Supplier ID</td>
<td>Supplier ID number</td>
<td>Numeric</td>
<td>Null</td>
<td>0012</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Normalized Relations**

- **CURRENT JOBS** (Current Job ID, Status, Invoice Number, Cost, Start Date, Est. Completion Date, Job Name, Company ID, Material ID)
  
  - Primary Key: Current Job ID, Foreign Key(s): Company ID, Material ID

- **CONTACTS** (Company ID, Company Name, Contact Name, Phone, Fax, Email, Address, City)
  
  - Primary Key: Company ID, Foreign Key(s): None

- **SUPPLIERS** (Supplier ID, Company Name, Contact Name, Phone, Fax, Email, Address, City)
  
  - Primary Key: Supplier ID, Foreign Key(s): None

- **MATERIALS** (Material ID, Name, Product Number, Pieces, Carton Cost, Cost per Piece, Supplier ID)
  
  - Primary Key: Material ID, Foreign Key(s): Supplier ID
Assumptions

- A contact can be assigned to more than one job.
- Each job uses only one supplier.
- Materials are ordered for specific jobs only so materials ordered can only be used for one job.
- Materials come from only one supplier for each job.
Personal Reflections

I chose to create a database for my thesis project because I found, through my classes at Western Michigan University, that it was the most intriguing CIS topic to me. While completing my class on database applications, I realized that I wanted a way, outside of the classroom, to flex my database skills. I chose to create an open source database because of the rising popularity of this type of software.

I believe that some people may assume that this project seems uncomplicated because it is, quite simply, a server with database forms. This couldn’t be farther from the truth. Because of the nature of open source software, I spent a vast amount of my time researching various problems with the server and office software. I was forced to read through hundreds of online forums for help from other users who were experiencing similar problems. Although it became frustrating at times, I preferred finding the solution on my own as opposed to being on hold for hours with a vendor of proprietary software.

I feel that completing this honors thesis has not only expanded my knowledge of databases but has also helped me to become more organized and responsible. I completed my honors thesis through a CIS independent study course and because of this I was able to set my own deadlines. Having this freedom forced me to make time management my top priority. I believe the skills that I learned about myself during this process will be very beneficial to an employer in the future.

While it was interesting to see the way that the open source community operates within itself, I think that I still enjoy receiving a software package that is as complete as possible. I enjoy troubleshooting, but for me, open source software provided more of that then I wanted to do. Also, I find that solutions to problems with proprietary software are more accessible to a
user than just searching blindly for a helpful forum. While I may not change my Microsoft ways, I think it was important for me to investigate open source software so that I can have a well rounded understanding of the software community as a whole.

All in all, this experience has been very worthwhile. I’m grateful for the Lee Honors College for pushing me to complete a project like this because it allows me to prove to myself and my professors that I am capable of doing it. I feel that completing an honors thesis at the undergraduate level is an advantage that not many other seniors will have. Hopefully this drive and motivation will help me to stand out amongst my peers in the corporate world.
Bibliography

  <http://www.mysql.com/company/>.


  <http://about.openoffice.org/index.html>.


  


  
  <http://www.php.net/>.

Appendix A – Viewing Database Information

The following appendices will use the Contacts Form in the database. The contacts form can be opened by clicking on the Contacts link in the OpenOffice.org Base main menu seen below.

Once opened, the form will populate itself automatically with the information currently in the database. The populated form is pictured below.
Appendix B – Adding Information to the Database

When the Contacts form is opened, it will automatically begin at the contact ID number 1. This field is set to automatically increment when new records are added. A new record can be added to the database by clicking the Add Record button highlighted below.

After completing all the fields for a new entry, the user must then click the save button highlighted below. This will save the new record into the database.
Appendix C – Deleting Information from the Database

In order to delete a record from the database, the user must have the record to be deleted displayed. The record can be deleted by clicking the Delete Record button. This button is highlighted below.

After clicking the Delete Record button, the following warning will appear.
Clicking Yes will change the record number back to 20, as seen in the following screenshot.
Appendix D – Searching the Forms

Every form can be searched individually by clicking the Find Record button. This button is highlighted below.

After clicking this button, the following screen will appear. This screen allows the user to select which text to search for, along with the option to specify which field to search in. For this example, the All Fields option was chosen.
After clicking the search button, the form generates the entry with the information the user searched for. This search result yielded the following form.

If the search box does not find any results, the following error message will appear.