Review of DBMS for Linguistic Purposes

Ferrara, Marisa ; Moran, Steven

Review of Database Management Systems (DBMS) for Linguistic Purposes

Marisa Ferrara and Steven Moran
Eastern Michigan University

1. Introduction

All linguists specializing in language description must confront the problem of how to digitally organize the data they have collected. With the new urgency of initiatives that seek to document endangered languages, the way in which data is organized and archived is of great importance: many formats become unreadable as technology advances and developers cease to support older versions. Standards in digital language documentation are important in another way: it is crucial that native communities be able to access material, regardless of the platform or software they are using. The E-MELD (http://emeld.org) initiative seeks to diminish the risk of language data loss and increase data accessibility by teaching linguists specializing in language description how to document their language in best practice format.

E-MELD recommends that linguists create a text file in XML markup as their archival copy, but archival formats are designed for preservation, not for modification. Therefore, many linguists use the formats that DBMS[*] and their interfaces (henceforth simply DBMS) produce as storage for their data. Despite interest in best practice initiatives, however, we know of no comprehensive review of best practice DBMS software used for language documentation. This project aims to develop an ongoing review that evaluates linguistic and non-linguistic DBMS according to best practice criteria and with a linguistic purpose in mind. In Section 2 we will discuss the project goals and our methodology for evaluating this software. In Sections 3 and 4 we present both linguistic and non-linguistic software that have been evaluated according to our criteria, and in Section 5 we discuss the software that we had not been able to fully evaluate at this time due to a variety of reasons.

2. Methodology

The goal of this project is to evaluate different tools linguists use for database management. The tools selected here are those that we have encountered through our work with the E-MELD project as well as those that have been recommended to us by various field linguists. If we do not mention a piece of software, it is not to discount it: we consider this an ongoing project. We would indeed appreciate any feedback regarding the criteria we have used and other software that should be reviewed.

We would also like to mention that the goal of our project is not to select the best software available, since any such recommendation is dependent upon the specific needs of individual linguists and projects. Our goal is simply to evaluate the software according to the criteria we have chosen and allow the user to decide which criteria are most essential to their needs. The following section describes how we established the criteria we have used on this project.

2.1 *Criteria*

We began our project by selecting criteria that we felt were essential for databasing linguistic information.[1] We distinguished between three types of evaluative criteria: General Information, Technical Information, and the Ability to Handle Linguistic Data.

2.1.1 *General Information*

[*] Please see Glossary in Appendix i for a full definition.
[1] Please see Appendix iv, v, and vi for a full list of the criteria we used to evaluate the software.

By "General Information" we mean information that a user may want to take into consideration when selecting software, such as what developer produced the software, which platforms the software runs on, what other software is needed, the price, and the licensing options available. We have also considered the ease with which the software can be downloaded*, how easy it is to use, and whether or not support for the software is available.

2.1.2 *Technical Information*

This section includes criteria that we found to be important for database software whatever its purpose, and includes database and network issues, programmability, and the ability to import and export data. We have considered as well those criteria used by other database reviews such as "Open Source Database Comparison" (Anonymous 2004) and "Questions to Help Evaluate Linguistic Tools" (BIFoCAL 2003b). One such criterion is database type. We have evaluated relational, XML, flat, and free-form databases for this review, and found that all three were able to handle linguistic data. We also found it critical to establish whether the database had a pre-defined, rather than user-defined, structure, as this could affect how easy it might be to enter data and add missing features at a later time.

We evaluated overall database stability with such criteria as ACID-compliance (*Atomicity, Consistency, Isolation,* and *Durability*, four criteria which are considered essential for database design by business professionals)* and data integrity, both of which ensure that should something go wrong, data loss will be kept to a minimum. ACID-compliance in particular is considered by business professionals to be essential when rating overall database stability; most commercial applications are therefore compliant in this regard. Maintenance of data integrity was deemed very important, as this rates how well the database constrains the user to follow good database design (for instance, there should be primary key constraints and child records should be updated when database changes are made).

We also considered whether or not multiple users could use the same database and whether a network connection and/or web access is needed to use the software. This, obviously, could be problematic when collecting data in the field. Likewise, we wanted to mention any programming interfaces or API available for the software, in case the user would like to extend the functionality of the program.

Because XML* is best practice for archiving linguistic data, the technical section considers the import and exportability of XML as well as of other formats. This is of particular importance for linguists since using data in other software applications as well as sharing data will be increasingly essential.

2.1.3 *Ability to Handle Linguistic Data*

The final section, Ability to Handle Linguistic Data, was a consequence of our work on the E-MELD project, and of course consideration of the work of other linguistic tool evaluations such as BIFoCAL (BIFoCAL 2003b). In it we address such issues as whether or not the software is designed specifically for linguists, Unicode* compatibility, special character input methods, and the ease of character input. We also address search functionality, whether or not our searches returned relevant information, and whether the software can search across fields. This last criterion is important since linguists often need to search for information contained in two or more fields rather than just one. We encountered this problem when we tried to search for tonal information and vowel information, which we happened to store in two different fields.

This section also evaluates other key features a linguist would need, such as the ability of the software to handle primary texts and interlinearize material, MDF (Multi-Dictionary Formatter)* options, and the ability to generate lexicons or grammars. These are not

* Please see Glossary in Appendix i for a full definition.

absolutely essential, but make documenting a language much easier. Whether or not the software has the ability to easily store any audio, video, or images through an easy-to-use interface has been noted in this section as well. We've also considered whether or not the user has the ability to add missing features, and how easy it is after such a change to update existing records.

Each piece of software also has specific negatives and positives. Where these were relevant, we have mentioned these, as well as the specific set of users we would recommend the software for.

2.2 *Test data*

Once we chose the criteria for evaluation, we tested each software program with original linguistic data. This data was provided by Steven Moran from field work undertaken during the summer of 2003. The data is from the endangered language Sisaala, Western [SSL] (Ethnologue 2004), henceforth Sisali. Sisali is a Gur language spoken in Northwestern Ghana (Upper West Region). There are approximately 6,000 speakers and the language is unwritten and previously not described. Typological characteristics of Sisali include SVO word order, left-headed NPs, and a contrasting high/low tonal system. The data for the project included a 20 entry subset of the Sisali lexicon[2], a common example of which is provided in Table 1. Each entry includes the following fields: ID, Form, Gloss, Grammatical Category (Gram Cat), Comment, Source, Reference, and Date. These are defined as follows:

ID:	The numeric value given to each entry in ascending order.
Form:	The phonemic form for an entry. This may be a word or phrase.
Gloss:	The form's English gloss.
Gram Cat:	The grammatical category of the linguistic form.
Comment:	Any comments written down by the field linguist.
Source:	The language consultant who provided the data.
Reference:	A numeric value which refers to the page number on which the entry can be found in the field notebooks.
Date:	Date on which the data was collected.

Table 1: Example of Linguistic Data

ID	Form	Gloss	Gramm Cat	Comment	Source	Ref	Date
2562	o fa ka poolla	sick; he was sick	vp:intrans:pst:3Psg		Cletus Basing	120	25-Jul-03

2.3 Software

As was mentioned above, the pieces of software that we chose to evaluate were either those we had encountered through the E-MELD project, those that had been referred to us by field linguists, or those that we found via the web. This produced a list of 14 software applications:

-Access 2003 (http://www.microsoft.com/Office/Access/prodinfo/default.mspx)
-askSam 5.1(http://www.asksam.com/)

[2] Please see Linguistic Data in Appendix iii for the full subset of data.

-Emdros 1.1.17 (http://emdros.org/)
-Excel 2003 (http://office.microsoft.com/home/office.aspx?assetid=FX01085800)
-eXist 1.0 (http://exist.sourceforge.net/index.html)
-FIELD beta (http://emeld.org/tools/field/beta/)
-FileMaker Pro 7 (http://www.filemaker.com/)
-Kura 2.0-1-2.1.2 (http://www.ats.lmu.de/kura/index.php)
-LinguaLinks Workshops (http://www.ethnologue.com/LL_docs/LingWksh.asp)
-MATES (no URL available at this time)
-MySQL (http://www.mysql.com/)
-PostgreSQL (http://www.postgresql.org/)
-Shoebox 5.0 (http://www.sil.org/computing/shoebox/)
-Word 2003 (http://www.microsoft.com/office/word/prodinfo/default.mspx)

Of this list of tools, we only managed to fully evaluate seven: Access 2003, Excel 2003, eXist 1.0, FIELD, FileMaker Pro 7, MySQL, and Shoebox 5.0. The remaining software could not be fully evaluated due to a variety of reasons. For example, some software we deemed to be so technical as to be usable by very few, other software promoted bad practice, and some software was still in development and we were therefore unable to evaluate in time. For more information about the software not fully evaluated and any problems we encountered, please see Section 5.

2.4 General problems

A variety of problems were caused simply by the structure of the data. Tone is a syllabic phenomenon, usually not a lexical one. Yet none of the systems allow you to mark tone in a separate tier, linked to the syllable. The system which comes closest to this ideal is FIELD, which treats tone as a separate phenomenon, but here tone is linked to the lexical entry, not the syllable. Likewise, with Shoebox, we found the only logical way to input tone was to use IPA characters in a "Tone" field, and even then it was difficult to break apart the morphemes.

In FileMaker Pro, Access, and MySQL we had decided to take advantage of the relational database structure and use link tables. Although it is easy to create these, constraining the values and ensuring that the correct entries are made in them is difficult in FileMaker Pro, and to a lesser extent, in Access. MySQL really constrains the user to good database design, but the user must either use the SQL command line (which requires knowledge of the SQL language) or implement an interface (pre-existing interfaces are available, or the user can program their own in a number of different languages). In the end we came to the reluctant conclusion that link tables may not be the best idea if ease of input is a top priority.

With regard to import and export, we found that format was crucial. In one set of data, for example, morphological boundaries had been indicated with colons and semi-colons: these caused problems when we tried to import them. It became clear that it was not possible to talk about ease of import without also considering what format the data was in.

We also considered how to input semantic fields in a systematic way. In the relational database structure we once again looked into link tables, but the time it took to input forms deterred us. Shoebox allows the user to simply insert text, but it does not constrain the user in any way to an existing set. We did find that FIELD worked particularly well for this, since the semantic categories can be chosen from menus. However, it generally takes considerably longer to input forms into FIELD than into something like Access or Excel. Overall, however, we found – unsurprisingly – that working with interfaces specifically designed for linguistic purposes on the whole made problems easier to solve.

We will now present the evaluated software in two categories, Linguistic DBMS and interfaces and Non-linguistic DBMS. Linguistic DBMS are those tools that have been developed specifically for linguistic purposes, whereas non-linguistic DBMS were developed for general purposes but can be customized for use with language data.

3. Linguistic DBMS and Interfaces

The following section discusses the software applications we reviewed which were developed with language data and a linguistic purpose in mind. There are obviously many advantages to using this software; however many linguists have chosen to use non-linguistic software instead. The reasons for this are various. Sometimes the software is less difficult to download or use, sometimes it is more user-friendly, and sometimes it is not as difficult to customize. The fact still remains, though, that linguistic DBMS and interfaces, despite their problems, ultimately provide greater functionality for linguistic purposes than their non-linguistic counterparts, and should therefore be strongly considered by any field linguist. In addition, this software is usually much easier work with, particularly at the beginning, than the non-linguistic software, since with non-linguistic software the user will be required to create an interface for their data before inputting.

We would also like to point out the distinction between linguistic DBMS and linguistic interfaces. Although we have used linguistic DBMS as the cover-all term, there is a distinction between these two entities. Linguistic DBMS are full database management tools designed for a linguistic purpose, such as Shoebox. Linguistic interfaces, on the other hand, provide a way to enter data specifically with a linguistic purpose onto a DBMS that may have a general purpose. FIELD is a good example of this, as it stores data in an Oracle database, yet provides an interface that makes it very easy to input linguistic data.

The software applications we chose to evaluate at this time are E-MELD's FIELD, and Shoebox, developed by SIL International. We also attempted to evaluate other software that is either currently in development (Kura, MATES, Emdros) or otherwise unavailable, but due to the problems encountered, we were unable to finish the evaluation of these products. For further information on the problems we encountered, see Section 5.

3.1 *FIELD*

FIELD (Field Input Environment for Linguist Data) is a web-based input tool developed by The LINGUIST List (http://linguistlist.org) as part of its NSF-funded E-MELD project. Designed by linguists and programmed by a computer scientist, FIELD was initially developed to input language data from Biao Min into the FIELD language database for endangered languages data. During development it became clear that FIELD would be useful as a generalized data input tool. Consequently, FIELD has continued to be developed and has now been used to input data from more than 12 languages.

3.1.1 *General Information*

FIELD is a web based input tool for entering linguistic data into the FIELD database. The FIELD web interface interacts through Coldfusion with an Oracle database (version 8.1.6.1.0) which houses the FIELD language database. The FIELD interface is open-source and free for use by researchers, and in theory could work with any full relational database, such as MySQL. New users must establish a login, through which they provide minimal information including their name and email address. Because this tool is web-based, it will run on any platform that supports the web browsers IE (5.0 or newer) or Netscape (7.0 or newer). At this time FIELD cannot be downloaded, though a stand-alone version is under development.

Support for FIELD is available via a freely downloadable tutorial and through email. There are as yet no support groups or forums for FIELD on the web. Its users are limited in number though the tool is growing in popularity.

3.1.2 *Technical Information*

FIELD is a web interface to an Oracle relational database that has a predefined structure for language data. Oracle databases are ACID-compliant and FIELD has rollback and update features. FIELD is only accessible through the web, so an Internet connection is mandatory for its use. Also, browser support is limited to newer versions of IE (5.0 or newer) and Netscape (7.0 or newer). It is most likely usable in any newer browser which is Unicode and XML compliant, but this has not yet been tested by the E-MELD team.

FIELD supports both collaborative teams and single users. The software is open-source, but no API is available at this time. FIELD does not now have SSL access, though it can support the protocol. An import XML feature is presently in testing, as is the ability to import Shoebox files and Microsoft Excel files. FIELD can, however, export XML and tab-delimited text format. When exporting XML, the user can presently choose to view their data through two predefined stylesheets, known as Standard and Report formats.

3.1.3 *Ability to Handle Linguistic Data*

FIELD is designed exclusively for linguists. It has been developed by linguists with the specific purpose of storing lexical data in best practice format; therefore, it was developed with full Unicode compliancy. FIELD also incorporates the GOLD linguistic ontology (Farrar & Langendoen 2003) into its language profile creation. Each user must create a language profile for each language that they enter data for. This has many benefits: for example, grammar sketches can be automatically generated and linguistic features can be searched across languages. It also means that closely related languages can reuse each others profiles.

FIELD uses Charwrite (http://emeld.org/tools/charwrite.cfm), a Unicode input tool for the web, to input IPA characters. Charwrite is a very easy tool to use and is accessed by mouse clicks on text boxes in FIELD web forms. It has two main kinds of input: if you double click on a text box, a clickable IPA chart pops up in separate window. By clicking on the characters in the IPA chart these characters are sent into a preview text box whose contents are then sent back to the text box in the FIELD web form. There is also a similar character recognition function: if you enter a character into the text box and then right click (Mac users: control-click), a clickable pop-up window appears with similar characters to the one the user has entered. For example, right clicking on 'a' returns the choices: ä, æ, ą, ɐ, ɑ, ɒ, ɒ̈, ʌ, à, á, â, ã, ä, å, æ, ā, ă, ą, ǽ. This makes inputting IPA characters and diacritics very easy.

FIELD also has the ability to create a user-defined keyboard. This can be used as a pop-up keyboard, keyboard shortcuts, or both. The pop-up keyboard is created by choosing characters from a large number of consonants, vowels, common orthographic symbols and diacritics. Descriptions are available for each character by moving your mouse over its glyph. The user then arranges the characters with diacritics for their individual pop up keyboard. The pop up keyboard then inserts data by mouse click. In the second option, keyboard shortcuts are created by choosing characters (and diacritics where applicable) and then assigning keystrokes to them. These features are extremely helpful for inputting data and have been made very simple to use. Charwrite is available for free download at http://cf.linguistlist.org/cfdocs/emeld/school/toolroom/software/charwrite-installation-instructions.html.

Another nice feature of FIELD is its comprehensive search facilities. There are two main searches, the *Browse* and the *Search Entries* functions. Browse and search options are populated through the language profile, so that users can automatically search on those

features from which they have created their language's profile. Searches can also be done with wild cards.

FIELD is primarily a lexical database and does not handle texts or interlinearization of texts, though these features are in development. However, example sentences can be entered with lexical items if the user chooses this option during the language profile setup. FIELD does not support MDF (Multi-Dictionary Formatter), and does not have the ability to store audio, video, or image files, though an interface to the audio-video annotation tool Elan is being built. On the other hand, FIELD can generate different styles of lexicons through its use of stylesheets, and a grammar generation feature is provided through the user's input of the language's linguistic features in the language profile. One of the major advantages of FIELD is that users can add new or missing features to their language profile and update their data with ease.

3.1.4 *Overall Assessment*

Pros:

FIELD's best quality is that it has been developed by linguists and is being continually expanded as typologically diverse languages are used with the tool: the system is under constant development, and is being modified regularly to handle more and more diverse data. The tool is simple to use and the interface's learning curve is not steep. Also, the tool's development will continue into the future.

Cons:

Since FIELD is only accessible via the web, users must have an Internet connection. Though the interface is easy to use, FIELD is still under development and is being continually updated. This can translate into errors. For instance, during our evaluation we came across two Page Missing errors and one ColdFusion error. When these occurred we hit back on the browser and resubmitted our data each time with success. Also, since FIELD has only limited import functions (many of which are still under development), adding large amounts of linguistic data is very time consuming. Though a quick, spread-sheet like input tool is being developed, it does not yet exist, and inputting data form-by-form into FIELD is often time-consuming.

Recommended for:

The linguist who would like to adhere to best practice and would like to be able to access and input their data securely via the web. Also for the linguist who would like to take advantage of simple character input and powerful searching options.

3.2 *SIL Shoebox 5.0 with Toolbox 1.2*

SIL's Shoebox, although currently unsupported, is still one of the most popular database management tools among linguists. It is designed entirely for language data and offers functionality that all non-linguistic database software evaluated here lack, such as a native environment for text interlinearization and analysis. It even works well with anthropological data. Toolbox 1.2 adds Unicode support to the software and Shoebox is able to export XML, leading it to better practice.

3.2.1 *General Information*

Shoebox is distributed by SIL and is available on both Windows platforms (3.1, 95 and later) as well as Mac OS 7.5 and later. It stores data in a marked-up text file, without extension, and the user can easily open the files in a text editor. At $19.95, it is very

reasonably priced when compared with other proprietary software. It is also very easy to download.

In order to get Unicode support for Shoebox, the user has to download Toolbox 1.2, which is available free from the SIL site. A Tavultesoft Keyman Keyboard must be downloaded (http://www.tavultesoft.com/keyman/) along with this. Since there are no pre-defined Keyman Keyboards developed for IPA in Unicode, the user will also have to download an evaluation version (free) of Keyman developer and develop their own keyboard.[3] Although this is somewhat complex, it does allow users to decide on their own keyboard shortcuts for the IPA symbols they will use the most. Once installed, they can use the Keyman Keyboard on almost all Windows applications that can handle Unicode, including Microsoft Office and the IE browser. If the user would prefer to extend an existing Keyman Keyboard, this may cause more problems as they may have to switch between keyboards. However, in our experience, we found the Keyman Keyboard we developed was one of the more easy to use tools for inputting IPA compared to the other options we tried in the Windows environment. Unfortunately, as we note in 3.2.3, the existence of a partially Unicode compliant version of the software does not mean that Unicode works everywhere in Toolbox.

Shoebox is fairly easy to use, though it requires some work: it is not very intuitive and that the user has to be prepared to do some work before they can access all its functions. Laura Buszard-Welcher reiterates this problem, stating, "Rumor has it that Shoebox gurus can get the program to do backflips; however for the average user, there seems to be a steep learning curve" (2003). Shoebox does have an excellent help function that was able to handle all the questions we had, but since Shoebox is unsupported and hardly any helpful information is available through a Google search, it might be best to take Buszard-Welcher's advice and "talk to other people who have used it" (2003).

3.2.2 *Technical Information*

Shoebox uses a text database that has a fixed format, but no predefined fields. Thus the user can name fields anything they like and use them to annotate data as they like. However, if the user would like to use the MDF formatter, they must use the pre-defined MDF fields, which are available online via a quick Google search. Since Shoebox is not a major database, it is not ACID-compliant, though it does have two rollback features. The *Undo* function allows the user to undo the last action and the *Undo All* function allows the user to clear all changes made to the record. Although there is most likely little damage the user could do to the data without trying, it's probably also a good idea to save continually to prevent data loss.

Shoebox is not collaborative, therefore no network connection is necessary. Also, the software is not open-source, so there are no programming interfaces or APIs available to extend the functionality of the program. The only way to import data into Shoebox is through a properly-tagged text file that uses something called a "change table". There are no clear instructions on how to create a change table, although there are a few available with the software. This means that unless a user happens to have exactly the right kind of data, they will not be able to import their material easily.

Shoebox is able to export in XML, which is a very nice function. The XML it exports is not standard, however: it produces a malformed XML document with the wrong header information. Once we replaced the header with a well-formed XML header[4], however, the XML document looked great and was also in a very intuitive format (tagged by the field labels). This is very different from FileMaker Pro and Access XML exports. Shoebox also gives

[3] An article by Lorna Priest provides full instructions on creating this keyboard. See References for URL.

[4] <?xml version="1.0" encoding="UTF-8" ?>

the user the option of exporting a file with field descriptions, saving a step should the user want to import their data into another application. Other export formats are RTF and text.

3.2.3 *Ability to Handle Linguistic Data*

Since Shoebox is designed for linguistics, its ability to handle linguistic data is superior in most ways to some of the non-linguistic software we have evaluated. And, now that it supports Unicode with Toolbox 1.2, it is software which can largely produce best practice documentation.

When searching, setting up a filter to return certain information in Shoebox is much easier than setting up a query or script in non-linguistic software (Access and FileMaker Pro, for instance), and with a little bit of trial and error the user can learn how to ask the questions correctly to get the information they are looking for. Unfortunately, the Keyman Keyboard does not work in the Filter dialog box, which means that the user must cut and paste Unicode characters in. The search dialog box, however, will not even accept the characters once they have been pasted in. This means that there are certain Unicode problems even with Toolbox. Finally, we could not figure out how to get the filters or searches to work across fields, although it might be possible.

One of the strengths of Shoebox is its ability to handle primary text information and interlinear glossing. It does take some patience to get the interlinear glossing to work if the user does not have experience working with the software, but with a little bit of time and effort Shoebox can handle the information better than any of the other software we have evaluated so far. That is not to say, however, that it does not have its limitations – other users have complained that it lacks a function "to link dictionary entries with morphemes in the texts" (Buszard-Welcher 2003). Shoebox also allows for MDF formatting, lexicon generation, and even grammar generation, although the last function was not tested during this evaluation. The user can even link .wav files to records and can save images, though the software cannot store video data.

3.2.4 *Overall Assessment*

Pros:

By far and away, Shoebox's best quality is that it was developed with linguistic data in mind. It can perform functions that linguists specifically need, and with practice the software can accomplish great things and make analyzing data much easier.

Cons:

Shoebox's biggest downfall is that it is no longer supported, so the user will have to live with any bugs they find. It is also unable to import XML, which would have been a nice feature for users who would like to use the primary text functionality of Shoebox simultaneously with the lexical functionality of another program. And, Shoebox — even with the Toolbox add-on — is not completely Unicode-compliant. Finally, the Shoebox environment is not very user-friendly, and it takes considerable practice to get the full functionality from the software.

Recommended for:

The linguist who would like to use the linguistic functions that Shoebox has to offer at the expense of a steep learning curve and less-than-perfect best practice.

4. Non-Linguistic DBMS

Because there are so many field linguists who have chosen to use non-linguistic software for databasing linguistic data, we decided to evaluate some of the top contenders among commercial and open-source projects. The greatest benefits these pieces of software have to offer field linguists are ease of use, customizable interfaces, and a large amount of support due to the software's wide application. Although the user will have to develop an interface before they can begin inputting their data (unlike linguistic DBMS), this could be an advantage for linguists who would like full control over their data. We decided to choose from these contenders only those tools that deal with two very important facets of best practice for linguistic data – Unicode compatibility and XML import and export.

Given these criteria, the software we have evaluated to this point include: Filemaker Pro, Microsoft Access, Microsoft Excel, MySQL, and eXist. Although Excel is not a DBMS, we decided to include it in our evaluation since it does provide a lot of the functionality needed and many linguists use it for their data. Also, Section 5 includes a list of those applications we decided not evaluate at this time.

4.1 *FileMaker Pro 7*

FileMaker Pro database software is one of the most popular for non-programmers, as it provides an easy interface and a lot of versatility for anyone wishing to database information. It is of course not linguistics-specific, but it is fully customizable and does provide basic functions that work well with lexical information. The newest version, FileMaker Pro 7, includes XML input and output as well as UTF-8 compliance, making it a good contender amongst the other software evaluated here. FileMaker Pro does have its limitations, however, as it cannot perform all the functions needed to make linguistic analysis easier. The following details our findings in evaluating the software.

4.1.1 *General Information*

FileMaker Pro is commercial software distributed by FileMaker, Inc. The software was evaluated on Mac OSX because it runs best in its native Mac environment. The data is stored in .fp7 files, which are proprietary, but the user can easily export to a variety of non-proprietary formats (see *Technical Information* below).

At $159.00 (academic version), FileMaker Pro is not particularly costly, and the lack of computer expertise needed for download and use nearly makes up for the cost. A free 30-day evaluation version download is available on the website as well. If problems are encountered, there are a variety of tutorials and books available online for free that can walk novices through setting up a database structure, all of which are extremely helpful since the FileMaker Pro *Help* function is not very extensive. Since the software is commercial and widely used, it was found that a quick Google search can provide answers to many questions as well. Technical support is available from FileMaker Pro, but it is somewhat costly ($3/minute US). There are various groups available online, however, that can answer questions for free.

4.1.2 *Technical Information*

Although previous versions of FileMaker Pro offered a basically flat database design, the 7th edition establishes itself as a fully relational database (Clyman 2004). It is fully customizable and therefore does not have a pre-defined database. In order to get the best functionality, it might be best to quickly read more about relational databases and consider a database design before getting started. Also, FileMaker Pro has very few data constraints, therefore users should be aware of the importance of primary keys and other database design issues in order to ensure data integrity. This is particularly true since once the user has established a system and entered data, it is a bit of a hassle to change the database

structure. Also, FileMaker Pro is fully ACID compliant and is therefore a basically stable environment to save data in. It also provides data roll backs and easy backup functions just in case the user accidentally deletes something that should not have been deleted.

Since FileMaker Pro 7 can be both single user and collaborative, a network connection is only necessary to work with the data if the user wishes to collaborate through FileMaker Pro Server or publish their data online. The server allows up to 350 simultaneous users, while regular FileMaker Pro can handle 5. SSL is available, but only through FileMaker Server.

FileMaker Pro only handles FileMaker Pro scripts as a programming interface for the search functions. The scripts are relatively easy to learn, but they can only do so much – other reviews have commented that Access' querying language is much better able to handle more complex database design (Engelberg 2000). Since the software is commercial, it is not possible to extend the program's built in functions, and no API is available.

The nice thing about FileMaker Pro is that even though it is proprietary, it allows the user to import and export data in a variety of formats. First and foremost of importance for best practice are the XML import and export functions. Although the user can import XML, the software requires that they have a database layout that matches the XML they import from. It is also not very easy to keep a relational database structure when doing this, so it is probably best to just use a flat database design. The XML export function also has its limitations, as the user can only export in a pre-defined XML layout that is not the most transparent – they will most likely need to simplify it or provide a schema in order to import it to another program. Although we did not test every single import and export function, the delimited text files are a definite plus and the Excel and Access imports function well. However, the HTML output requires some fiddling with.[5]

4.1.3 *Ability to Handle Linguistic Data*

FileMaker Pro is not designed for linguistics; therefore, any ability that it has to handle linguistic data must be customized by the user. This is of course one of its strengths since with a little patience a great interface can be developed that will make entering data incredibly easy and specific to the user's needs.

Many versions of FileMaker Pro have claimed to be Unicode compatible, but it appears that with the 7th edition the developers repaired some of the previous bugs with importing and exporting Unicode. We did not encounter any problems at all with the Unicode encoding in the data we tested. Inputting IPA requires either the Mac character map or keyboard input through the character map's shortcuts. This can be time-consuming, and since the shortcuts are not user-defined it may require a little more effort than with other input methods.

FileMaker Pro's Find function uses an easy-to-learn syntax, Script Maker, to define searches based on regular expressions. For less complex searches, this should be fine, although there are a few drawbacks as FileMaker Pro can not handle more complex searches as easily as Access can, and the user can not save a search for later use. Also, FileMaker Pro makes it very difficult to search across fields.

The major drawback to FileMaker Pro is that it can not handle two very important functions for linguistic data analysis: There is no function that will allow the software to handle primary texts, and according to one linguist, "FileMaker Pro is very difficult to work with for parsing and interlinearizing texts and is mostly unsatisfactory for these tasks" (Sprouse 2003). On the bright side, finding different layouts for the information is quite easy in FileMaker Pro, and although there is no MDF-type formatter or grammar generation built in, the extensive export formats should make it easier to work with another tool. Another bonus

---

Please see Glossary in Appendix i for a full definition.

[5] Please refer to the table in Appendix v for more details on import and export formats.

is that the user can store audio and video data in fields in the database, but they will need an annotation tool to align texts to A/V material and transcribe them.

4.1.4 *Overall Assessment*

Pros:

FileMaker Pro's best feature is its great user interface and ease of use. For anyone who does not have a technical background but would like to develop an extensive database for their linguistic data, FileMaker Pro offers great functionality despite the fact that it is not designed for linguists. Also, its customizable interface allows the user to develop an interface to suit their own needs.

Cons:

Obviously the biggest con for FileMaker Pro is that it can not handle everything required for linguistic analysis, such as interlinearizing or even databasing texts. Therefore, although it is easy to use, it does not provide some of the nice extra features that Shoebox does to make the analysis that much easier. It is also very difficult to encode tone information in a way that makes sense and is easy to use (link tables will require extensive setup and may take longer for input), other than through phonetic IPA characters. Another major problem with FileMaker Pro is that its XML import and export functions do not allow the user to define a particular schema and will most likely require extra work before they can be used by another application. And finally, FileMaker Pro does not constrain the user to provide a Primary Key field or even follow good database design, which can have negative consequences in a relational database system.

Recommended for:

Any linguist who is not comfortable with a computer and would sacrifice automatically-generated linguistic analysis for ease of use.

4.2 Microsoft Access 2003

FileMaker Pro and Microsoft Access are very similar programs that aim to accomplish the same basic functions. However, there are some differences between the two that could affect which software is best suited for the user's purposes. The general consensus on the basic difference is that Access is better suited for more complex databasing and search functions, whereas FileMaker Pro provides a nicer interface and works well for more simple databases (Engelberg 2000). In evaluating this software, we found that this was generally the case, since although Access provides a less intuitive interface, it does allow for more complex databases and searches, and better practice in implementing them.

4.2.1 *General Information*

Microsoft Access is packaged with Microsoft Office and is therefore usually available on any Windows machine. It runs on Windows 2000 and XP as well as Mac OS X, and it was tested on Windows XP for this evaluation. The reason why Access 2003 rather than XP was used is due to the fact that the 2003 edition adds Unicode compatibility and better XML output, which is essential for best practice.

At $189.00, the software is a little bit more expensive than FileMaker Pro, although it now comes shipped with Office 2003 on certain Windows machines. It is also incredibly easy to download. The user interface is not as easy to customize as FileMaker Pro, however it is not by any means difficult to do. Should the user encounter any problems, a free online tutorial is available with purchase, and the *Help* function answers a lot of questions. As with

FileMaker Pro, a quick Google search can answer a lot of questions as well. Technical support is available but expensive at $35 US.

4.2.2 *Technical Information*

Access is a relational database, although a flat format is possible. Also, since it is customizable, there is no pre-defined layout and time should be spent thinking through a functional database design before implementing and entering data. Access is also fully ACID-compliant, and it has a rollback feature that allows the user to return to a previous state. Another major advantage to Access is that it constrains the user to good database design in a relational structure, unlike FileMaker Pro – primary keys are required, and link tables are much easier to set up and constrain.

Access can be either collaborative or single user, although the user will need to establish a server in order to share the database (Microsoft SharePoint is recommended on the site). We did not evaluate the software's collaborative functionality; therefore, it is unknown how well it works. A network connection is unnecessary unless the user intends to use SharePoint, which also offers SSL access.

The programming interface for Access is VBA (Visual Basic for Applications), and the search function uses this query language, which according to reviews, is less intuitive but more powerful than FileMaker Pro's ScriptMaker (Engelberg 2000). There are a variety of free tutorials online that can help the user learn VBA, and since the user can save queries, Access' search function is much more useful than FileMaker Pro's (assuming that a more complex level of searching is required).

Access does allow for XML import and export, and it allows the user to define a schema for both (Access can also provide its own schema for the standard XML output). This gives the user much more control over outputs than FileMaker Pro, although it does require some level of comfort with the languages to get the imports to match up with database fields. Also, Access has a wide range of import and export formats including various non-proprietary formats.[6]

4.2.3 *Ability to Handle Linguistic Data*

Access is not designed specifically for linguists, and is therefore going to require some customization prior to data entry. However, with a little bit of patience, a database system that can handle linguistic data efficiently can be created.

Microsoft Access claims to be fully Unicode compatible, and although we did not encounter any problems with the data that we used, there have been reports of the encoding not exporting correctly in certain formats. We developed and used a Keyman keyboard[7] to input the IPA symbols, which was easy to use once it was developed (although set-up does require time, in the end it is worth it since the only alternative would be to use the character map).

Although it takes a bit of time to get used to VBA, the Access query language, it offers a wide range of functionality that can be very useful. Technically, the user can even get it to search across fields, although we had trouble implementing this. The search return relevance depends on how well the user develops the query, but once the user gets used to the language it should be relatively easy to write queries to suit their needs.

Access does not have a lot of the functionality needed to deal with linguistic data beyond lexical information. Using the export facility and another piece of software, it is possible to develop a lexicon, but grammar generation and interlinear text handling is

[6] Please refer to the table in Appendix v for more details on import and export formats.
[7] See section 3.2.1 for more information on implementing a Keyman Keyboard.

impossible. Also, Access allows the user to save A/V material, but only by saving a URL to the file in a specific field. Therefore, at least one other program will be needed to complete a linguistic analysis.

4.2.4 *Overall Assessment*

Pros:

Microsoft Access 2003 is more powerful than FileMaker Pro but less user-intuitive. If the user is looking for ease of use with almost full database functionality, Microsoft Access would be a good bet. It constrains the user to good database design much more than FileMaker Pro and can also output an XSD schema and XSL stylesheet with any XML document. Another plus is that Microsoft Access has a default input much like Microsoft Excel, which quite a few linguists use, but has added database functionality that Excel can not handle.

Cons:

Access is not as user-friendly as FileMaker Pro, and it requires motivation on the user's part to learn how to take advantage of all it has to offer (which, for a linguist, might not necessarily be a lot). Much like FileMaker Pro, it is difficult to enter tone information in an easy to use (and easy to create) way with link tables, so the user will have to really consider customizing a full interface in order to make input easy (we simply decided to use IPA tone characters).

Recommended for:

Linguists who need a more complex database system and can take the time to learn how to manipulate Access' functionality to suit linguistic purposes.

4.3 Microsoft Excel 2003

Microsoft Excel is today's most popular spreadsheet program, with an estimated 90% of the spreadsheet market (Walkenbach 2004a). Excel has replaced Lotus 1-2-3 as the de facto for spreadsheet programs and comes prepackaged on some Windows machines. With its spreadsheet input interface, we found Excel to be a good tool for creating lexicons. Its consistent structure allows for consistent exporting of data, but its cell structure does not provide a good interface for texts.

4.3.1 *General Information*

Microsoft Excel is packaged with Microsoft Office, therefore it is usually available on many Windows machines. Excel 2003, as well as Microsoft Office editions in the 2003 release, require the Microsoft Windows 2000 Service Pack 3 or later, or Windows XP or later (recommended). For this evaluation Excel 2003 was tested on a Windows XP machine. Originally the data was stored in Excel 2002 but Excel 2003 offers extended XML compatibility and therefore was chosen for evaluation.

At $229 for new users, the software is more expensive than Microsoft Access, FileMaker Pro, and Shoebox. There is however an upgrade price of $109 if other versions have been previously purchased. Microsoft also has special prices for other licensing (including for academic purposes), and Excel comes shipped with Office 2003 on certain Windows machines. Microsoft offers purchasing through mail order as well as through an incredibly easy download.

The user interface is similar to other Microsoft Office products and is limited in its customizability. The main workspace is a typical spreadsheet with cell formatting and cut and paste features. Tool bar options include adding, removing, and arranging buttons and

notebooks can be arranged in windows (tiled, vertical, horizontal, and cascade). For those used to Microsoft Office products, Excel will be familiar and easy to use.

Should the user encounter any problems, a free tutorial is available with purchase, and the *Help* function answers a lot of questions, though it is often tricky to install. When working with Excel 2002 and accessing the *Help* function we were prompted to insert the Microsoft Office Installation CD for Help Installation Support. We were also prompted for an Internet Connection when searching with the Answer Wizard in Excel 2003. A quick Google search will answer most questions and there are various discussion groups available for query (it is often the case that similar problems have already been encountered by other users). Technical support is available but expensive at $35 US.

4.3.2 *Technical Information*

Excel is a spreadsheet program that saves data in the Microsoft propriety file format .xls by default. Since Excel is not a database program the user must not implement a database structure before use. However, consideration should be taken when designing the spreadsheet layout. Careful mapping of columns and rows may save the user a lot of trouble when exporting to other file formats such as XML. Since Excel is not a database, ACID-compliance was not established during our evaluation. Rollback features are limited to the *Edit* > *Undo* function and it is recommended that users save their data periodically and in chronological editions in case problems arise.

Excel can be either collaborative or single user. For collaborative use the user must *Share a Workbook*. This is set through the *Tools* menu under *Share Workbook* on the *Editing* tab. Excel also has a feature that allows changes by more than one user at the same time, although the file must be on a network location accessible for intended users (this should be on a shared network folder, not a web server). We did not evaluate this feature.

For searching in Excel the user must use the *Find* or *Find and Replace* function. The user can search across Sheets or Workbooks and by Rows or Columns or a combination of these. Searches can also be case sensitive or restricted to entire cell contents.

Microsoft Office provides drivers that allow Excel to access numerous data sources (Microsoft SQL Server, Access, dBASE, Microsoft FoxPro, Oracle, Paradox, SQL Server, Text file databases, and Third-party providers). Excel also has a feature to export data to an Access database and Microsoft makes it easy to work between the two programs.

Excel does allow for XML import and export and it also allows the user to define a schema for both. Excel 2003 can take advantage of well-formed XML files and Microsoft has also defined an XML spreadsheet format designed specifically for Excel worksheets. When opening an XML file that references a stylesheet the user has the opportunity to open the file without applying any stylesheet, or the user can apply a specific stylesheet when the file references more than one. Excel 2003 opens and saves properly structured XML files, it can create web queries (a query that retrieves data stored on an intranet or Internet source) and it allows users to save entire workbooks as XML spreadsheets. When importing or exporting XML, Excel validates against an XML map when the *Validate data against schema for import and export* is enabled in the *XML Map Properties*. When exporting XML data, Excel applies the following rules:

-UTF-8 encoding is used to write the data
-All namespaces are defined in the Root element
-Comment nodes are not preserved
-Empty items are not created when blank cells exist for an optional element
-Empty items are created when blank cells exist for a required element

-Excel overwrites existing namespace prefixes. The default namepsace is assigned a prefix of ns0. Successive namespaces will be designated ns1, ns2, to ns<*count*> where <*count*> is the number of namespaces in written to the XML file.

Excel also has a wide range of *Save As* file formats including various non-proprietary formats. Please note that purchasing Microsoft Office 2003 may not give users access to new XML features in Excel 2003. These features are available only in the 'Professional' edition of Microsoft Office (Walkenbach 2004b).

4.3.3 *Ability to Handle Linguistic Data*

Excel is not designed specifically for linguists and therefore requires some customization prior to data entry. With patience however, Excel can be a very useful tool for storing lexical data and for formatting data for dictionary creation.

Microsoft Excel, like Access, claims to be fully Unicode compatible, and although we did not encounter problems during our evaluation, there have been reports of the encoding not exporting correctly in certain formats. Because we developed and used a Keyman keyboard to input IPA symbols in Microsoft Access, we chose to limit inputting IPA symbols to the *Insert* function in Microsoft Excel, which is the software's prepackaged default for character insertion. The *Insert* > *Symbol* function is limited in its usability. The user may insert characters but can not define keyboard shortcuts as in Microsoft Word. A *Recently Used Characters* function provides an easy to click on list of the most recently used characters, but the user must always go through the *Insert* and *Symbol* functions to access this list. We recommend inserting commonly used symbols into a cell from which they can be cut and pasted easily.

4.3.4 *Overall Assessment*

Pros:

Microsoft Excel is a good program for housing lexical data in a structured manner that allows for consistent exporting of data in numerous formats. More recently Excel has been part of a limited number of software packages that are Unicode compliant, and therefore has been chosen by a number of linguists intending to follow best practice by using Unicode. Excel also provides an input environment that allows for fast data entry. Unlike many other software programs, inputting data in Excel is limited to the typing speed of the user, though this may take tricks like the following. For instance, with the *Global Replace* function all unique character pseudonyms can be quickly converted to their IPA symbol. For example when evaluating Excel we used capital E for ɛ, which was quickly converted to ɛ with the Find and Replace feature.

Cons:

As a spreadsheet program Excel does not handle texts. Cell formatting is at first tricky and Microsoft's typical autocorrect features can be annoying. For those with previous knowledge of Microsoft Office products, Autocorrect and Autoformat options should most likely be switched (or defaulted) off before inputting linguistic data.

Recommended for:

Linguists who wish to quickly digitize lexical data to be later exported into XML.

4.4 MySQL 4.1

MySQL (http://www.mysql.com/) is the world's most popular Open Source database server. Written in C and C++ by Michael (Monty) Widenius, MySQL is a SQL* database server that is both multi-threaded and multi-user, and is considered very fast and robust (Active-Venture 2003a). A relational database management system, MySQL is distributed by the MySQL AB commercial company, founded by MySQL developers. Because it is a database server, it should be noted that though it comes with an interface, it is limited in its functionality. A web interface built into HTML, such as PHP, will allow users to adapt and manipulate the full features available from MySQL. This is specifically true for linguistic data input.

We would also like to mention that some linguistic interfaces have been developed to run on the MySQL and PostgreSQL databases, including Kura and Emdros. Unfortunately, we have not yet been able to evaluate either software, but we encourage users to look into these programs if they are considering using MySQL or PostgreSQL.

4.4.1 *General Information*

MySQL is dual licensed and is available as open source/free software under the GNU (General Public License) license or under a commercial license ($495) distributed by MySQL AB. MySQL database is available on a multitude of platforms (often requiring native or other threads): AIX 4.x, 5.x, Amiga, BSDI 2.x, 3.0, 3.1, 4.x, DEC UNIX 4.x, FreeBSD 2.x, 3.x, 4.x, HP-UX 10.20, 11.x, Linux 2.0+, Mac OS X, NetBSD 1.3/1.4, Novell NetWare 6.0, OpenBSD > 2.5, < 2.5, OS/2 Warp 3, SCO OpenServer, SCO UnixWare 7.1x, SGI Irix 6.x, Solaris 2.5 and above, SunOS 4.x, Tru64 Unix, Windows 9x, Me, NT, 2000 and XP.

Although MySQL is very easy to download, we recommended that the diligent user spend some time reading through MySQL's free online tutorial. Particularly relevant is chapter 2, *Choose which distribution to install*. As was the case with our evaluation, we first installed the current stable version of MySQL (4.0) and were left later to upgrade to a Unicode compliant version. At the time of this paper, MySQL 4.1 is in a development release series, yet it is the first Unicode compliant version. If the user has already installed MySQL database server, they may wish to upgrade to 4.1, even though it is currently in beta status.

The online tutorial provided by MySQL clearly documents current issues that should be read before and during installation and when upgrading. Overall we found the online searchable tutorial is invaluable, particularly since it continually incorporates user comments. Also, installing MySQL requires other software that is documented in the tutorial.

We installed MySQL 4.0 on a PC with Microsoft XP, but in addition we needed MyODBC (an ODBC*) and IIS* (attainable through *Adding/Removing Windows Components*, which prompted us for the Windows XP installation disc) for installation. Technical support is available through MySQL AB and pricing is available at the following url: http://www.mysql.com/support/pricing.html. There are numerous discussion groups on the web, and a Google search returns relevant queries (a search for 'MySQL General Information' returns 1,800,000 hits). MySQL also has built in help functions available with the *--help* or *-?* functions.

4.4.2 *Technical Information*

MySQL is a relational database management system and it uses SQL as its query language. SQL is the most common standardized language and is defined by ANSI/ISO standards (Active-Venture 2003b). The MySQL database is not designed for linguistics; therefore, a database structure must be defined before entering data into tables. Because it

* Please see Glossary in Appendix i for a full definition.
* Please see Glossary in Appendix i for a full definition.

is relational, the database is very powerful and adds speed and flexibility. However, a relational database design is harder to implement than a flat database design -- we suggest that computer savvy individuals or language documentation projects first invest time in learning about relational database design and key definitions, such as 'unique identifier' and 'link tables'.

ACID-compliant transaction functionality is now a standard feature for MySQL (MySQL 2004) and with a transaction storage engine (InnoDB) the software also provides rollback and crash recovery capabilities. MySQL is both collaborative and single user and a TCP/IP connection was needed to run the software on our Windows XP machine, even though we connected locally. Web access is available through a number of web programming languages that allow the user to access the database through code embedded in HTML. PHP is often used to create such interfaces for MySQL through web pages, and is also very popular because it is open source and very accessible.

SSL access exists, and there are many accessible APIs, "one or more for almost every programming language" (Dev Shed 2004). Also, because MySQL is so popular and because it is open source software, there are countless add-on tools and applications that have been built on the MySQL database.

MySQL is very robust and one of its most unique features is the community of open source software developers and the countless applications that continue to be created by such individuals. If a GUI-type interface was not available for importing XML data, one was then built (Database Journal 2004). Or if someone wanted to export XML data from their database, it had already been encountered and answered (ZDNet UK 2004). Of course MySQL provides SQL commands for importing and exporting data, for example the *mysqlimport* data import program. Such commands will most likely take some fiddling with to adapt them to user specific needs, so we also suggest searching the web for free tools developed by other users.

4.4.3 *Ability to Handle Linguistic Data*

MySQL is not designed specifically for linguists. It is interesting to note however, that though XML is currently best practice for archiving linguistic data in text format, this allows the user limited functionality with the data. Therefore many linguists are using DBMSs for their working format. This has many benefits including complex querying of the data.

As we found out through multiple installations, MySQL is Unicode compliant as of version 4.1 (currently being released in development series, beta). When using the DOS shell to start and access the database, we found it impossible to input special IPA characters. Also, we have not found a good GUI for inputting special characters (although it is very likely that one exists). Simply put, the user must create an interface (or find one that is already built) to insert special characters into a MySQL database. Through our work with Coldfusion and Oracle for the LINGUIST List, we know it is possible to insert special characters into our database through the web, using web browsers with Unicode support. Though we did not have the time, we plan to create and test PHP forms to input and retrieve Unicode data into and from the MySQL database. This will also allow the user to craft their own linguistic specific searches, which should allow users to take full advantage of MySQL ability to search across fields.

Unfortunately MySQL does not handle primary texts or interlinerazation of texts because it is a database and such features would need to be developed for linguistic-specific use. This also means that MySQL does not support MDF, lexicon, or grammar generation without designing and implementing software for these specific purposes (or finding someone else who already has). Audio/video/image storage is also limited to links in fields associated with entries and is dependent on playing or viewing software.

4.4.4 *Overall Assessment*

Pros:

MySQL is the leading open source DBMS with a huge following. There are an estimated four million installations and there have been up to 27,000 software downloads per day (MySQL 2004). This has created an invaluable amount of free application-specific software, created and distributed by loyal users on the web.

Cons:

MySQL is a database with a limited GUI. Users need to be aware that an interface (most likely a web interface) needs to be created to get full functionality of the database in regards to important linguistic aspects like special character input, lexicon and grammar generation. This can be done in many different web programming languages, such as the open source PHP.

Recommended for:

Computer savvy individuals and large projects that have members with computer expertise. Also, those willing to program their own interfaces.

4.5 eXist 1.0

Many consider XML databases to be the newest best practice (Frieb 2003), and since they seem to be a great way to store data in best practice format and utilize that data once it is stored, they may make for a great linguistic database. Tamino is the best-known XML database and is becoming extremely popular, although with a price-tag of $45,000 it is a bit out of range for our purposes here. After reading Werner Frieb's 2003 article "XML Databases Compared", we decided to go with his recommendation and evaluate eXist, an open-source XML database developed by Wolfgang Meier and available free online.

Before we go into the details of eXist, we would like to clarify that eXist does not give the user the ability to work with data or create XML files. It simply serves as a program that can store the XML files that the user has developed in other applications and search over them in an intelligent manner. Therefore, we do not recommend eXist as a first step to databasing language information; we think that it does provide a nice second step into best practice from a non-best practice application that has the ability to export in XML (Excel, Access, FileMaker Pro). It is also a relatively secure way to store data in case anything should go wrong with a primary version.

4.5.1 *General Information*

eXist is distributed under the GNU Public License and is therefore available free for use and development online. It is platform independent, although it has only been tested on Linux and Windows 2000, XP, and XP Server. It stores XML files in larger XML files that define the database, and does not require any other software for use.

Although it is very easy to download, in order to use it the user must learn XQuery, a query language that works on XML documents. The language is not difficult to learn, though it does require that the user become familiar with it in order to get any functionality from the program. Free tutorials for both eXist and XQuery are available online to walk the user through the program. There is also support available from a variety of Google groups and a SourceForge (http://sourceforge.net) mailing list.

4.5.2 *Technical Information*

eXist is an XML database that has a pre-defined structure. However, the XML documents the user imports into eXist do not have to conform to a specific schema other than one the user chooses to define. It does allow for rollbacks as well, though it is not fully ACID-compliant. It can be used by a single person or a collaborative group, although a network connection is necessary for collaboration and it does not offer SSL access.

To use eXist the user must learn XQuery, which will allow them to develop queries to filter and search over the data. Since eXist is open-source and written entirely in Java, if the user would like to extend the software a full API is available online. eXist only imports and exports in XML, therefore the user must have already created the XML documents he or she needs in order to use the database.

4.5.3 *Ability to Handle Linguistic Data*

eXist is of course not designed for linguistics, and therefore has limited linguistic functionality when compared to Shoebox and FIELD. However, it is Unicode compatible, so any Unicode data that is already in the XML documents the user has created will be safe in eXist. The search dialog box can even handle Unicode characters, so using XQuery to get data is considerably easy. And, eXist has the ability to search across fields, assuming the user acquires the proper syntax to develop the query.

eXist does not have any linguistic function other than storing linguistic data in a safe, best practice-compliant format and providing a nice interface to search over the data. The advantages of this are that the user can use another, less-than-best practice software to input their data, if they choose, and then move the data over to an XML database to comply with standards. A word of warning must be issued, however, that it is always best practice to use a best practice format *first*, since there is a danger of data loss or difficulty whenever conversion is done. However, if the user would rather stick with the input method they are using, and it has the ability to export in XML, then eXist may be a good choice for archiving data.

4.5.4 *Overall Assessment*

Pros:

The biggest advantage to using eXist is that it offers a best practice format for archiving existing XML files and performing queries over collections. It is also fully Unicode-compliant, open-source, and free to use.

Cons:

As we have mentioned, eXist is not a data entry database, and therefore necessitates the use of other software to create files that can export in XML. It also requires some technical abilities to learn and use XQuery, and difficulties or data loss during conversion may happen (although we did not encounter any such problems).

Recommended for:

The linguist who prefers to use a less than best practice format and then convert their data into XML. Also, for the linguist who would like to archive their data in a secure format and use the powerful XQuery language to develop search and filter functions.

5. Other Software

There is a wide range of other software that we began to evaluate but did not finish due to the fact that they either did not match our criteria or we encountered problems we were unable to solve over the course of this project. Should we solve any of the problems or if new versions address our criteria, we will add them to this evaluation at a later time.

5.1 *Software that is too technical*

Since we are gearing this project to software to be used by linguists specializing in language documentation, once we encountered a piece of software to be too technical we decided to leave it out of the evaluation. This was the case for two programs we began to evaluate, *Emdros 1.1.7* and *PostgreSQL 7.4*.

Emdros (http://emdros.org/) is a database engine for linguistic analysis and research, and it is also able to database analyzed and annotated text. Since it is one of the few database software available for purely linguistic purposes, we wanted to be sure to include it in the discussion. According to the website, its primary target domain is linguistic analysis on all levels, including morphology, syntax and discourse analysis. It is also designed for publishing data and text processing. However, Emdros is not stand-alone software – it requires the development of some kind of interface, which will then allow the user to use a MySQL or PostgreSQL database. Due to the fact that programming an interface is most likely not something that a linguist specializing in language documentation would like to do in their spare time, we decided to leave Emdros out of this evaluation. However, we encourage anyone interested to visit the site, as it seems that Emdros is a tool worth looking into if the technical aspects do not scare the user off.

PostgreSQL (http://www.postgresql.org/) is considered by many to be the most advanced and best open-source database system around. It is more highly developed than MySQL, and is fully customizable, although it is not specifically designed for linguistics. PostgreSQL has two major downfalls, however, that made it incompatible with our evaluation. First of all, its native environment is Unix, therefore the user must download an environment that simulates Unix (such as CygWin) to use the software on a Windows system. The download and setup of such a program is not very easy to work with. The other downfall is that PostgreSQL only has one GUI, which is extremely difficult to setup through CygWin. Due to these technical problems, we decided to restrict our evaluation to MySQL since it works much more nicely on Windows and Mac. But, if the user is comfortable with a Unix system and is considering open-source database software, they may want to look into PostgreSQL as an option, particularly since it offers quite a bit of support for open-source software.

5.2 *Software that is Bad Practice*

Although there are many linguists who still use software to manage their data that is not best practice, we decided to leave this software out of the evaluation due to the fact that we felt it best not to encourage linguists to use such software for their data. For example, we decided not to evaluate Microsoft Word since there are already reviews which expound how terrible it is as a tool for linguistic data, such as Jeff Good's "Microsoft Word: A Review of it as a tool for digitizing linguistic data" (2003).

We also decided not to evaluate AskSam (http://www.asksam.com/) for these reasons, even though AskSam is definitely better practice than Microsoft Word. AskSam is commercial, free-form database software that some linguists use for data management. It has an interface much like Microsoft Word, but with a database background that makes it very easy for people who do not want to deal with learning new interfaces. However, AskSam is not Unicode compatible. There are rumors that a version to be unveiled next year will have Unicode compatibility, however, as it stands now, we feel it best not to evaluate the software.

5.3 *Software still in development or unavailable*

Some of the software we found seemed to be perfect for the project, but upon closer inspection had much more practical problems that we could not solve during the timeframe given for this project. For instance, we were able to find a paper by Holub and Míka (2001) that discusses an "experimental linguistic database system" called MATES. Although we did not contact the researchers, we could not find any other information about this database available online, and were left to conclude that it must still be in development.

Another piece of software which we unfortunately had to leave out of this evaluation is Kura (http://www.ats.lmu.de/kura/index.php), a multi-user open-source linguistic database developed by Boudewijn Rempt and maintained by Peter Bouda. Kura has a nice website and is available for download. However, installation of Kura on both Windows and Mac platforms was incredibly difficult due to the fact that other software is required. After quite a few days of installation troubles, we finally contacted the developer for help. He replied immediately saying that he could help us, but was unfortunately not able to answer our questions in time for us to install and evaluate the software. If we can re-establish contact with the developer, we will of course evaluate the software at a later time. Also, if the user is comfortable in a Unix environment, they may have fewer troubles installing the software than we did, therefore we encourage all interested individuals to visit the website to find out more.

LinguaLinks Workshop is the newest tool that SIL is developing as a linguistic DBMS. Given the information on the website, it seems to be a great tool that extends Shoebox and is available with a variety of other software that linguists will be sure to use. However, we were unable to obtain a review copy at this time, but we will be sure to have a review of the software in the next few months as an extension of this project

6. Conclusion

Of the fourteen software applications that we began our survey with, we were only able to fully evaluate seven. However, we have managed to create general criteria focused on language data that can be applied to both linguistic and non-linguistic software. Although we can not recommend any one specific software, we hope that our criteria will help field linguists choose software that specifically meets their needs. Our evaluations will be disseminated through the E-MELD School of Best Practice Toolroom under the Software heading, and they will also be published online on the E-MELD site. And, as we have mentioned, we consider this project to be ongoing and we welcome any suggestions, comments, or reviews.

References

Active-Venture. 2003a. "The Main Features of MySQL". Retrieved July 13, 2004 at http://mysqld.active-venture.com/Features.html .

Active-Venture. 2003b. "Overview of the MySQL Database Management System". Retrieved July 13, 2004 at http://mysqld.active-venture.com/What-is.html .

Anonymous. 2004. "Open Source Database Software Comparison". Retrieved June 1, 2004 at http://www.geocities.com/mailsoftware42/db/ .

BIFoCAL. 2003a. "Software functionality for non-technical users". Retrieved July 1, 2004 at http://faust.linguistics.berkeley.edu/~jcgood/bifocal/SoftwareDims.html .

BIFoCAL. 2003b. "Questions to Help Evaluate Linguistic Tools". Retrieved July 1, 2004 at http://faust.linguistics.berkeley.edu/~jcgood/bifocal/SoftwareQuestions.html .

Buszard-Welcher, Laura. 2003. "Shoebox: A review of it as a tool for digitizing linguistic data". Berkeley Initiative for Computer Assisted Linguistics (BIFoCAL). Retrieved June 1, 2004 at http://faust.linguistics.berkeley.edu/~jcgood/bifocal/ShoeboxRev.html .

Clyman, John. 2004. "FileMaker Pro gets even better". *PC Magazine*. Retrieved July 1, 2004 at http://channelzone.ziffdavis.com/article2/0,1759,1550706,00.asp .

Database Journal. 2004. "Importing Excel, Access, or XML data into MySQL". Retrieved July 13, 2004 at http://www.databasejournal.com/sqletc/article.php/3095621 .

Dev Shed. 2004. "MySQL Programming". Retrieved July 13, 2004 at http://www.devshed.com/c/a/MySQL/A-Technical-Tour-of-MySQL/9/ .

E-MELD School of Best Practice. 2004. Retrieved July 1, 2004 at http://www.emeld.org/school .

Engelberg, Miriam. 2000. "Choosing between Microsoft Access and FileMaker Pro". Retrieved June 30, 2004 at http://www.techsoup.org/howto/articlepage.cfm?ArticleId=207&topicid=6 .

Ethnologue. 2004. "Sisaala, Western: a language of Ghana". Retrieved July 1, 2004 at http://www.ethnologue.com/show_language.asp?code=SSL .

Farrar, Scott and Terry Langendoen. 2003. "A linguistic ontology for the semantic web". GLOT International 7(3). 97-100. Retrieved July 9, 2004 at http://emeld.org/documents/GLOT-LinguisticOntology.pdf .

Frieb, Werner. 2003. "XML Databases compared". Retrieved June 21, 2004 at http://www.studierstube.org/world/xml_databases_compared.html .

Good, Jeff. 2003. "Microsoft Word: A Review of it as a tool for digitizing linguistic data". Berkely Initiative for Computer Assisted Linguistics (BiFoCAL). Retrieved June 30, 2004 at http://faust.linguistics.berkeley.edu/~jcgood/bifocal/WordRev.html .

Holub, Martin and Pavel Míka. 2001. "MATES – an experimental linguistic database system". *Proceedings of the IRCS Workshop on Linguistic Databases*. Retrieved June 1, 2004 at http://www.ldc.upenn.edu/annotation/database/papers/Mika_Holub/21.2.mika.pdf .

MySQL. 2004. "MySQL Database Now Provides Full Transaction Support". Retrieved July 13, 2004 at http://www.mysql.com/news-and-events/press-release/release_2002_11.html .

Nerbonne, John. 1998. "Introduction to John Nerbonne (ed.) *Linguistic Databases*". Stanford: CSLI. 1-12. Retrieved June 1, 2004 at http://odur.let.rug.nl/~nerbonne/papers/intro-db.pdf .

Priest, Lorna. 2003. "Building Keyboards with Keyman 6.0". *SIL International*. Retrieved June 23, 2004 from http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&item_id=KeymanIPATutorial .

Rempt, Boudewijn. 2002. "Kura". Retrieved June 1, 2004 at http://www.ats.lmu.de/kura/manual.pdf .

Sprouse, Ronald. 2003. "Filemaker Pro: A review of it as a tool for digitizing linguistic data". Berkely Initiative for Computer Assisted Linguistics (BiFoCAL). Retrieved June 30, 2004 at http://faust.linguistics.berkeley.edu/~jcgood/bifocal/FileMakerRev.html .

Walkenbach, John. 2004a. "The Spreadsheet Page". Retrieved July 1, 2004 at http://www.j-walk.com/ss/excel/

Walkenback, John. 2004b. "Excel 2003 Review". Retrieved July 1, 2004 at http://www.j-walk.com/ss/excel/xl2003.htm .

Webopedia. 2004a. "Database Management System". Retrieved July 6, 2004 at http://www.webopedia.com/TERM/D/database_management_system_DBMS.html .

Webopedia. 2004b. "SSL". Retrieved July 6, 2004 at http://www.webopedia.com/TERM/S/SSL.html .

ZDNet UK. 2004. "Creating XML from MySQL as easy as PI". Retrieved July 13, 2004 at http://insight.zdnet.co.uk/software/developer/0,39020469,2112200,00.htm .

Glossary

ACID-compliance: According to a website discussing open source database comparisons, the term ACID is an acronym for *Atomicity, Consistency, Isolation,* and *Durability*, four criteria which are considered essential for database design by business professionals (Anonymous 2004). The author goes on to discuss each criterion in detail:

> 1. Atomicity is an all-or-none proposition. Suppose you define a transaction that contains an UPDATE, an INSERT, and a DELETE statement. With atomicity, these statements are treated as a single unit, and thanks to consistency (the C in ACID) there are only two possible outcomes: either they all change the database or none of them do. This is important in situations like bank transactions where transferring money between accounts could result in disaster if the server were to go down after a DELETE statement but before the corresponding INSERT statement.
>
> 2. Consistency guarantees that a transaction never leaves your database in a half-finished state. If one part of the transaction fails, all of the pending changes are rolled back, leaving the database as it was before you initiated the transaction. For instance, when you delete a customer record, you should also delete all of that customer's records from associated tables (such as invoices and line items). A properly configured database wouldn't let you delete the customer record, if that meant leaving its invoices, and other associated records stranded.
>
> 3. Isolation keeps transactions separated from each other until they're finished. Transaction isolation is generally configurable in a variety of modes. For example, in one mode, a transaction blocks until the other transaction finishes. In a different mode, a transaction sees obsolete data (from the state the database was in before the previous transaction started). Suppose a user deletes a customer, and before the customer's invoices are deleted, a second user updates one of those invoices. In a blocking transaction scenario, the second user would have to wait for the first user's deletions to complete before issuing the update. The second user would then find out that the customer had been deleted, which is much better than losing changes without knowing about it.
>
> 4. Durability guarantees that the database will keep track of pending changes in such a way that the server can recover from an abnormal termination. Hence, even if the database server is unplugged in the middle of a transaction, it will return to a consistent state when it's restarted. The database handles this by storing uncommitted transactions in a transaction log. By virtue of consistency (explained above), a partially completed transaction won't be written to the database in the event of an abnormal termination. However, when the database is restarted after such a termination, it examines the transaction log for completed transactions that had not been committed, and applies them.
>
> (Anonymous 2004)

API: This is an abbreviation for *Application Program Interface*, which is "a set of routines, protocols and tools for building software applications" (Webopedia 2004c). They can also be used to better understand the software or extend the software's functionality.

DBMS: This abbreviation stands for Database Management System, which is "a collection of programs that enables [the user] to store, modify, and extract information from a database" (Webopedia 2004a). Although this is a general term, all the software we are evaluating within this paper are Database Management Systems in that they allow the user to store and work with their data.

IIS: An acronym for Internet Information Server. IIS (often pronounced "eyes") is a web server part of the Windows NT server.

MDF: This is an abbreviation for Multi-Dictionary Formatter. This tool allows the user to create and format a dictionary output easily from any text, provided the text has MDF-compliant tags. SIL's Shoebox comes with an MDF tool built in.

ODBC: An abbreviation for Open DataBase Connectivity, a standard database access method developed by the Microsoft Corporation. The ODBC inserts a database driver as a layer between an application and the DBMS, allowing the two to theoretically communicate transparently between different platforms and DBMSs. Both application and DMBS must be ODBC compliant.

SQL: This stands for Structured Query Language. Both an ANSI and ISO standard, SQL is a programming language for querying, updating and managing data.

SSL: This abbreviation stands for *Secure Sockets Layer*, a protocol for "transmitting private documents via the Internet" (Webopedia 2004b). SSL provides a secure way to transfer documents and information when using a collaborative tool (such as FileMaker Server or Microsoft Server). This is considered essential by business professionals when looking for database management software, and should also be considered by linguists if they would like to collaborate through the software.

Star rating:

★	Low
★★	Medium
★★★	High
★★★★	Very High

Unicode: According to E-MELD, "Unicode is an international character encoding standard used for plain text representation that has a standardized way of representing characters in all major writing systems of the world. The inventory of characters covered by the standard continues to grow; it has the potential to standardize codes for approximately one million characters. Unicode is the standard upon which many current fonts, keyboards, and software are based. For more detailed information, consult the latest edition of the Unicode Standard, which is available online from the Unicode website (http://www.unicode.org) and in print.(Anderson, 2003: 1) Also, see the E-MELD pages on Unicode (http://cf.linguistlist.org/cfdocs/emeld/school/classroom/unicode/index.html)" (2004).

XML: According to the E-MELD website, this stands for Extensible Markup Language, which "defines a standard way of encoding the structure of information in plain text format. It is an open standard of the World Wide Web Consortium (http://www.w3.org) that is based on extensible tags (extensible meaning that they are not pre-programmed, but can be defined by the creator). XML is currently considered best practice for the archival encoding of textual data, because it does not depend upon any particular software, and can be formatted through an XSL Stylesheet to be displayed in almost any format (including html, .txt, .doc). For more information see the E-MELD pages on XML (http://cf.linguistlist.org/cfdocs/emeld/school/classroom/xml/index.html)" (2004).