2 The Development of BugsCEP

This chapter provides a summary of the developmental work undertaken in the creation of the BugsCEP database and its program components. It also provides some basic background information that is a prerequisite for understanding certain more technical aspects of the system. Detailed descriptions of data and interfaces can be found in Chapter 3, and more specific details of the BugStats and BugsMCR components are to be found in the subsequent chapters.

2.1 Introduction

Programming is an undervalued activity in many areas of work outside of those where its application is most obvious – such as modelling, computer science and software development. Although the *data* is what most would regard as the core of any database, it is of limited worth without the software tools to enter, manage, retrieve and query it. In the case of BugsCEP, which is a comprehensive research and teaching software package build around a database, there is a considerable amount of coding, interface construction, query and report design built on top of the actual database. If one ignores the database structure, controls and properties and just examines the surrounding amount of physical (i.e. programmed or written) information it equates to over:

> 30 840 lines, 135 448 words, or over 892 834 characters
> (excluding spaces = 6.6 characters per word on average)

This is the equivalent of over 330 pages of text at 400 words per page. When the numerous rewrites, experiments and adjustments are accounted for, this forms the bulk of the spent time behind this thesis. Word counting the actual data is difficult, but it can reasonably be estimated to over 1 100 000 words. This represents over 2 500 pages of text that has either been converted from earlier versions of the database or entered by one of the current or previous developers. Approximately 60 % of these words is part of the biology and distribution data, a fact which clearly illustrates the importance of modern reference data in the database.

2.2 Database and Software Background

2.2.1 Relational database design

There is considerable variation in the use of the term 'relational database'. The original, and in some eyes official definition is complex and relies heavily on an understanding of set theory and logic (see Codd, 1970, for the initial definition). A simple layman's definition is chosen here: a relational database is where similar data are stored in matrices (tables) which are linked to each other by rules (relationships) governing their "...derivability, redundancy, and consistency" (Codd, 1970, out of context). This terminology contrasts with some more formal definitions, which will not be discussed here in order to avoid confusion[i]. The software used to construct and maintain a relational database is referred to as a 'relational database management system' (RDBMS, or RDMS). There is some argument as to which software packages may be considered as RDBMS's, but for the sake of simplicity MS Access, the database management software used to create the BugsCEP database, will be considered as an RDBMS in this thesis.

A relational database is then, a database designed so that there is no ambiguity over access to data items, no duplication or redundancy, and where data items are linked through a logical system of primary keys and indices which also enforce integrity rules on the data. (Redundancy is the repetition of identical data in a table – in other words, the inclusion of more data than is strictly necessary). By

[i] Readers are directed towards the Internet for thorough descriptions of relational database concepts. http://en.wikipedia.org/wiki/Relational_database is probably a good starting point.

following these guidelines a database can be constructed that is robust and secure in terms of read/write/delete access, and where relatively simple query expressions can be used to retrieve any combination of linked (related) data items. The process of reorganising data structures into relational database form is known as *normalisation*[ii].

Quaternary palaeoecology and environmental archaeology data are by their nature multi-proxy, consisting of several measurable quantities that can give a greater insight into past climates, environmental change and human impact together than they can independently. They are also chronologically and spatial extensive, and thus form a multi-dimensional data domain which must be normalised into an efficient data structure if it is to be accessed efficiently. Although the domain is complex, it is logically structured, and thus lends itself relatively easily to normalisation. The Coleopteran data form just a small part of the data-space occupied by the wealth of proxy data sources available, and multi-proxy databases such as SEAD (Buckland *et al.,* 2006) are considerably more complex that BugsCEP.

Database tables are linked by 'one-to-many' relationships where the data held in the *primary key field* are unique for every record in that table (the 'one' side of the relationship). The field with the same name in the table on the 'many' side of the relationship may contain several records where this value is identical, and is referred to as the *foreign key*. In the structural example shown in Figure 2.1, taken from the more thoroughly normalised SEAD environmental archaeology database, the use of one-to-many relationships can be seen for a number of fields. In the table *tblTaxaInsectGenera*, which stores the scientific names of insect genera, the beetle Genus 'Carabus' could have the unique identifier GenusID=3 (i.e. in the single record where the field 'GenusID' has the value '3', the field 'GenusName' contains the value 'Carabus'). The one-to-many relationship with table *tblTaxaMasterInsects* thus allows for several records in the latter where GenusID=3, the remaining fields of which hold data (or references to data) on all the species within the Carabus Genus[iii].

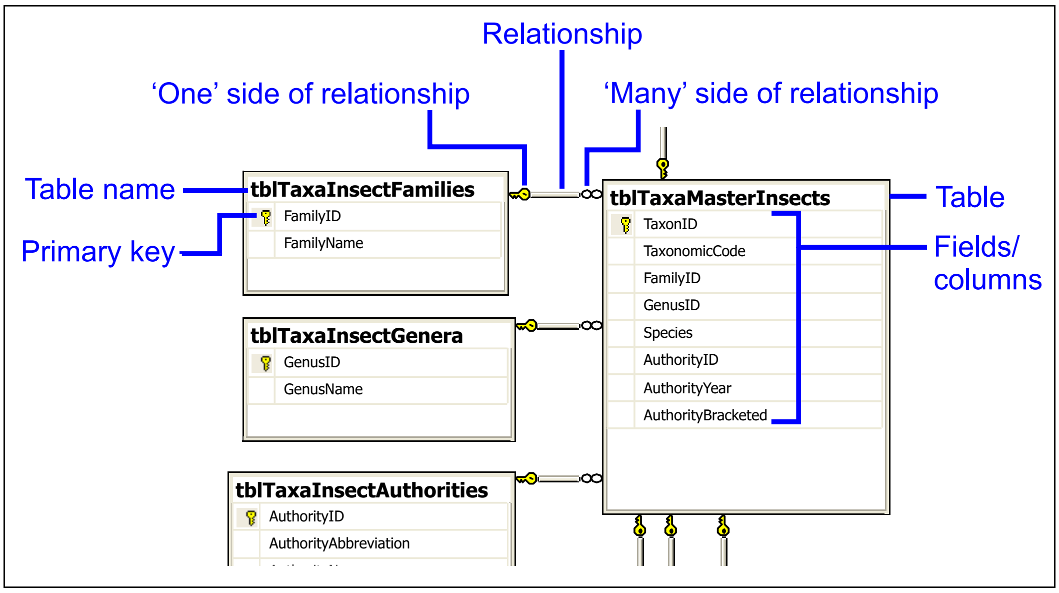


Figure 2.1. Relational database terminology, and the normalized taxonomic index structure of the SEAD database (Buckland *et al.,* 2006).

[ii] See e.g. Gifford *et al.* (1996), although any practically any relational database design book will do.
[iii] Note: This structure diagram was created with MS Visual Studio, using the SQL Server Express Edition management console, which can produce more illustrative structure diagrams that MS Access – the RDBMS used to create BugsCEP. The BugsCEP database structure has been recreated in SQL Server to improve presentation, but as yet only runs through MS Access.

Relational integrity enforces rules for the relationship between a primary key field ('one' side of the relationship) and its use as a foreign key in another table ('many' side of the relationship). Once enforced, there can be no data items in the foreign field that are not present in the key field, and it is a useful system for ensuring data validity. In MS Access this function can be left turned off if necessary, thus allowing for the easier import of old data. It can then be activated once the imported data have been verified (and will produce an error if the data violate the relationship criteria), a function which proved very useful during the transference of data from the previous version of Bugs. With referential integrity enforced, changes to any data that are in a primary key field will cascade to all related tables, and thus further ensure data validity.

The relational structure also forms a framework for the construction of database (SQL) queries, which are the primary method of data retrieval, with the logic of the relationships between tables providing pathways for the recovery of any data item. RDBMS packages such as MS Access and MS SQL Server provide intuitive interfaces to assist when creating queries, and write most of the SQL code themselves – a considerable improvement of older database management software. These interfaces, however, are difficult to use without a knowledge of database design, and are replaced in BugsCEP with custom built user interfaces.

2.2.2 The BugsCEP database structure

As described earlier, BugsCEP uses the frontend-backend application structure common to many database applications, as illustrated in Figure 2.2. This setup has many advantages, but perhaps the most important two are:

1. Increased data security – the potential for data loss and file corruption are reduced by keeping the data and program separate. This is particularly useful due to a number of bugs in MS Access, which could potentially cause this to occur.

2. Updatability and ease of distribution – the frontend can be updated without requiring a new copy of the backend. Likewise, updates to the data can be distributed independently of the program files. This means that smaller files can be distributed for updates. It also allows databases to function more easily in a multi-user environment, where a copy of the frontend is installed on each client workstation, but only one copy of the backend database is installed on a central server. In this way all users have access to the latest data[iv].

MS Access creates single file databases – that is to say that all the tables, queries, interfaces etc. can be stored in a single file[v]. In the BugsCEP system this results in two files, one each for the frontend and backend. The frontend also utilises several small external library files, which fulfil a number of common Windows tasks such as browsing/opening/saving files, and provide access to a number of MS Visual Basic and data management programming language components[vi].

[iv] Note that BugsCEP is not specifically developed to run in a multi-user environment, even though previous versions were. This is a result of the increased complexity and reliance on external library files in the new version, and the fact that there has not been time to put in place the necessary program parts to ensure reliable multi-user access.

[v] With the extension '.mdb'

[vi] Specifically, MS: VBA; Access Object Library; DAO; stdole; ADO; Common dialog library; Excel; Scripting library; Common controls; Hypertext help library.

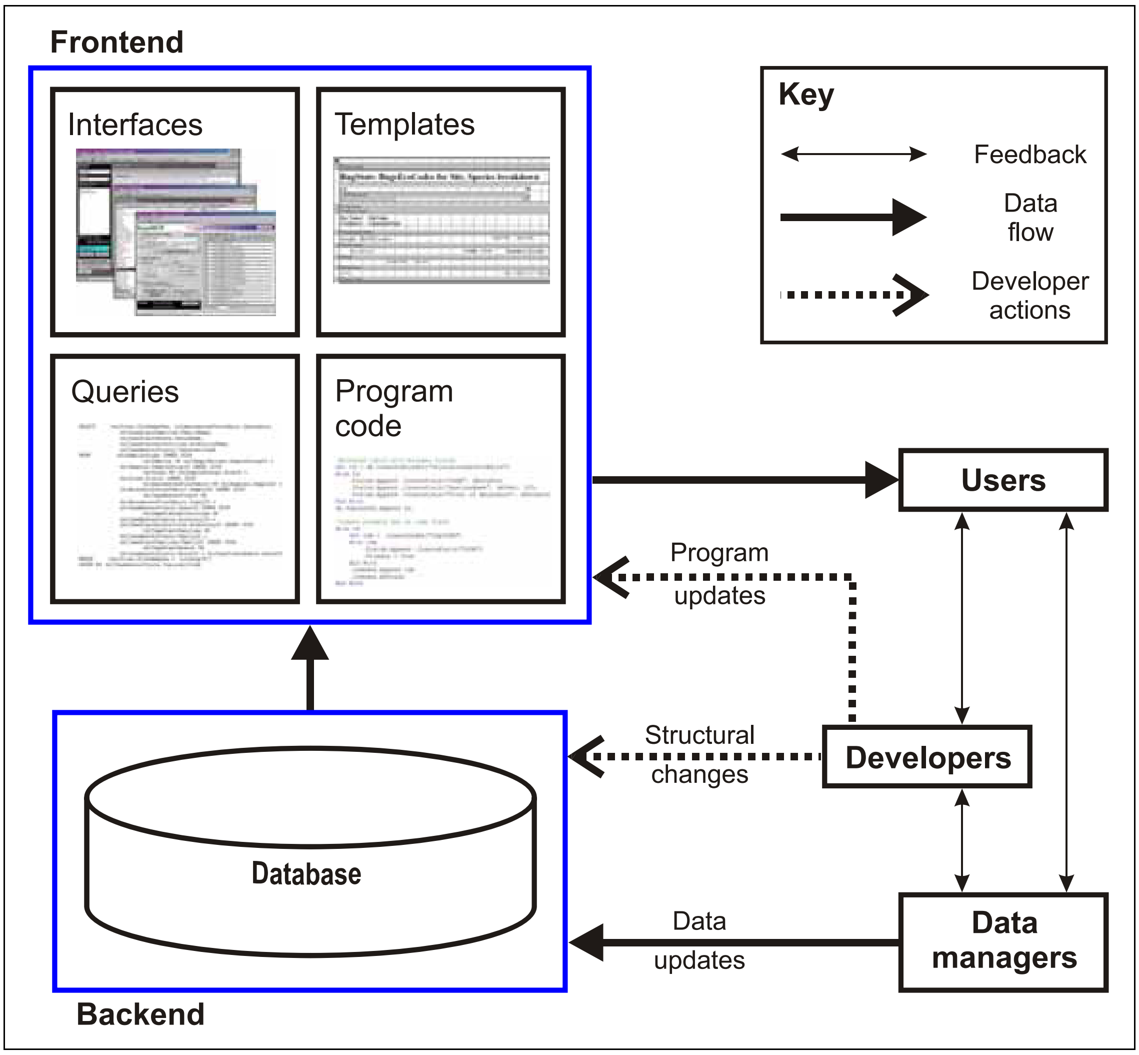


Figure 2.2. BugsCEP program frontend-backend structure, showing user, developer and data manager interaction.

The frontend contains only temporary data tables, along with settings for the current installation, and uses links to retrieve data from the backend. The structure of the backend is a highly, but not fully, normalized relational database as shown in Figure 2.3 (see later in this chapter for a discussion of some of the problems with this structure). The figure shows the relationships between the tables that store the BugsCEP data. An important feature of BugsCEP is its storage of modern ecology and distribution data – rather than just fossil data. This is illustrated on Figure 2.3 by grouping the tables that hold modern or fossil/site based data. Palaeoentomology works by comparing fossil assemblages with the habitats, biology and distribution of modern individuals in order to reconstruct past environments. As a user extracts information for a particular species found in a particular sample, the relationships enforced between the tables ensure that the modern reference data for that species can be obtained directly, and that any changes cascade automatically between the tables.

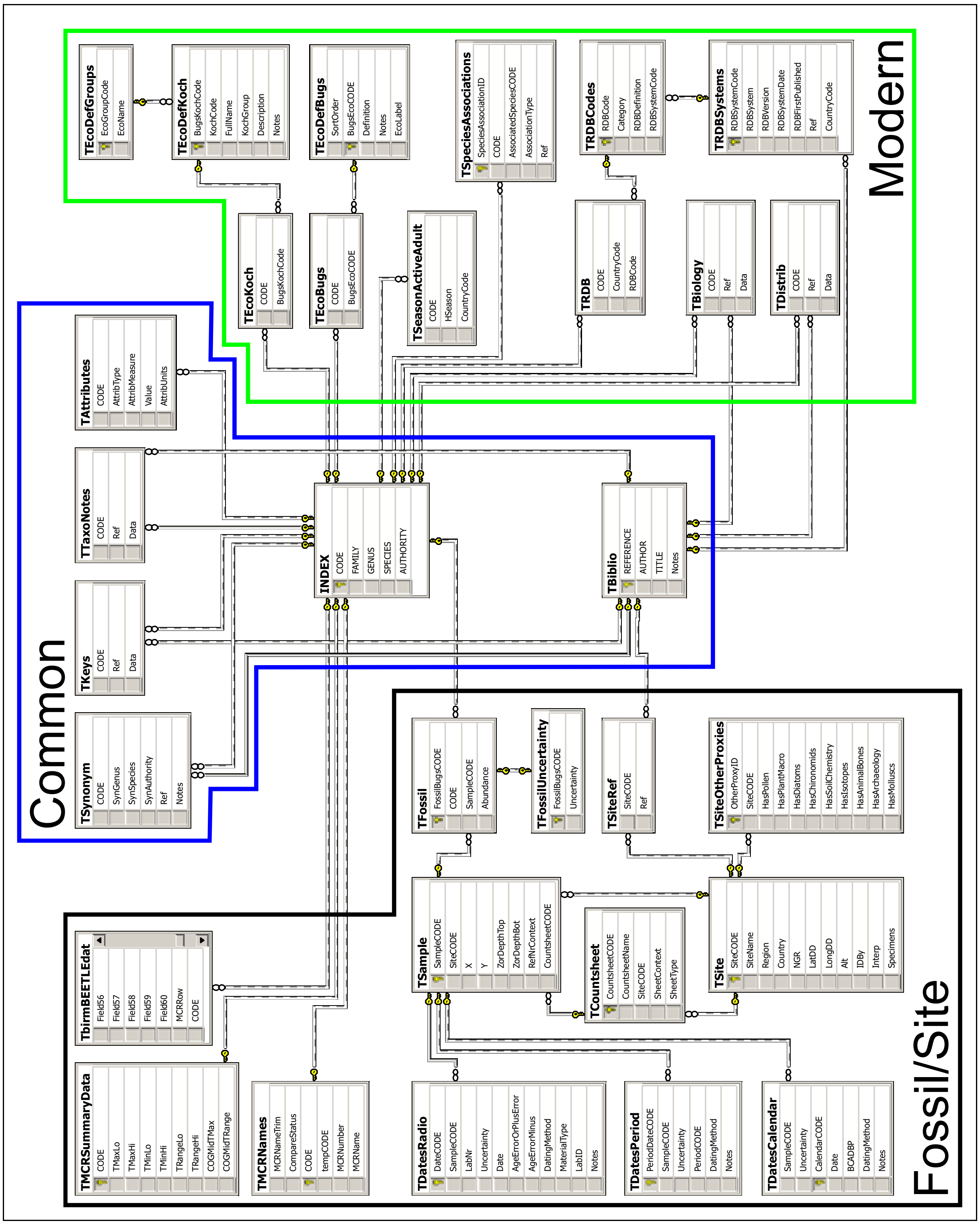


Figure 2.3. BugsCEP backend (database) structure. Boxes represent data tables with lists of their fields enclosed. Lines represent relationships between the tables (see Figure 2.1 for explanation). Note that a number of lookup and reference tables have been omitted to improve clarity. The three table groups, or data areas, Common, Modern and Fossil/Site are explained in Chapter 3.

2.2.3 Overview of BugsCEP software features

BugsCEP contains a large number of features, which are summarized in Table 2.1, and described in detail in the subsequent sections and chapters.

Table 2.1. Summary of BugsCEP program features.

Species data retrieval	*The main interface allows you to browse by species and read or extract:* *- published ecological information with references* *- published distribution data with references* *- synonyms* *- RDB - Red Data Book status (only UK so far, but facilities for more)* *- known (Quaternary) fossil record, sites and dates (if present)* *- coded ecological summary (Koch (1989-92) and internal Bugs EcoCodes)* *- taxonomic notes and limited size attributes*
Search by habitat (and more)	*Lists of species that match specified criteria can be obtained (ecology codes, RDB and biology and distribution text). These lists may be exported with a variety of information - including ecology and distribution, EcoCodes and references. BugsCEP can produce summarised lists of sites which contain the selected species.*
Site and collection data storage	*Users can store their collection/sample data as any number of countsheets per site. Site summary information, including latitude and longitude can be stored, and output in reports. BugsCEP can create a number of reports that summarise/list the ecological and distribution data for all species found at a site, with references. Species lists and countsheets can also be imported from MS Excel files. Lists are automatically sorted into taxonomic order.*
MCR Climate reconstruction	*Mean summer and winter temperatures can be reconstructed from species lists, sample by sample, using the MCR method (Atkinson et al., 1978). BugsCEP will produce simple thermal diagrams in MS Excel, along with exporting the raw data and sample thermal envelopes (climate space maps). It also has the facility to show which species could theoretically survive in any given temperature range, effectively allowing users to predict changes in species distributions with climate change.*
Environmental reconstruction	*Habitat summary diagrams can be created, showing the changing ecological implications of species between samples. A variety of statistical treatments are available – including standardization, abundance weighting and ln(n+1) transformation. Summary reports can be created to help in the analysis of these diagrams sample by sample. Correlation coefficients can be calculated to assess the (dis)similarity between all samples at a site.*
Reporting and exporting	*A variety of time saving reports can be created and exported, including:* *- summary data for any taxon* *- all information for species found at a site, with references* *- abbreviated forms of the above* *- summary information for all sites which include specific species* *- sample by sample, species by species breakdown of ecological implications of species found at a site* *- MCR climate reconstructions* *- Bugs EcoCode environmental reconstructions* *- exported countsheet in MS Excel format*
Bibliography	*A comprehensive bibliography of over 3 300 papers on beetle biology, distribution and fossil records is included in BugsCEP.*

2.3 Developmental Strategy

Database design projects have a greater chance of success if a design strategy is arrived at before the project is undertaken. The strategy may include goals with different levels of priority, but should always encompass an analysis of the system to be replaced, and the goals to be achieved. The goals which orientated the development of BugsCEP are to a large part synonymous with the aims of this thesis, and are summarised below in order of priority, with comments on the implementation and realisation of these aims.

2.3.1 Primary developmental aims

1. Develop a new, relational database, version of the Bugs software.
2. Develop a system for the graphical comparison and interpretation of fossil/modern insect faunas (with the working name: BugStats).
3. Implement and improve the MCR climate reconstruction method in Bugs, i.e. port to Windows and improve the MS-DOS software, and make MCR generally more accessible to all (with the working name: BugsMCR).

Implementation of Primary Aims

1. A completely new database structure, as described above, and new interfacing software were created to form BugsCEP. To avoid the inevitable legacy problems common when trying to adapt existing software to new purposes, no program code was retained from the old version. The software components are described Chapter 3 of this thesis.
2. & 3. BugStats and BugsMCR were developed as individual programs in order to achieve a more rapid release as test versions, and enable the use of the encompassed methods in teaching. Once fully functional, they were then integrated into the main BugsCEP program, and phased out of use as standalone programs. Such a strategy was greatly eased by the use of object orientated and modular programming, and only a minor amount of revision was needed in order to get the components running together. Redundant program code, that is to say code that performed essentially the same task but was found in more than one program component, was rewritten as common subroutines/functions with the appropriate structure to enable it to be called from the original locations. This also allowed for easier debugging, despite the inevitable increase in complexity of the subroutines. The user would notice no change, but future developers would be extremely grateful for having to only modify the program in one location rather than several. BugStats and BugsMCR are outlined in Chapter 3, and described in detail in chapters 4 and 5 respectively.

2.3.2 Secondary aims

4. Test the system on own data.
5. Test the system on published modern and fossil data.
6. Formally publish and distribute the BugsCEP software, and present it in a medium which allows for extensive discussion.

Implementation of Secondary Aims

4. The fossil Coleoptera remains from two sites were examined: a peat sequence from northern Sweden (Hemavan), two archaeological sequences from southern Sweden (Lockarp). In addition, a small modern fauna from Greenland was re-examined (GUS), and an attempt was made to assess the last 20 000 ^{14}C years of climate change, as reflected in the faunas from ^{14}C dated samples stored in BugsCEP. Preliminary data were also examined from lake Njulla in northern Sweden. Two further Swedish sites were examined, but not included in this thesis due to either lack of sufficient fossils or time (Bymyran and 'David's Bog'). The case studies are presented in Chapter 6.
5. Two published datasets were chosen to provide a test of the tools contained in BugsCEP for aiding the interpretation of modern and fossil beetle assemblages, the limited amount of time available restricting the number of datasets that could reasonably be examined. The reanalysis of a recent Finnish study of Carabidae across forest-farmland transects, and a study of a 140 000 year long peat sequence from eastern France, are presented in Chapter 6 section of this thesis.

6. The http://www.bugscep.com/ website provides an effective medium for the distribution and passive marketing of the database. Additional active marketing, in the form of posts on newsgroups and emails to established contacts announcing the release have been undertaken to a limited extent. The publication of this thesis allows for a more comprehensive discussion of the data and functionality of BugsCEP than individual articles or a manual would allow. With hind sight, an active marketing initiative is necessary to ensure that people are aware of the software and its possibilities, and attain the widest possible user base.

2.3.3 Development platform

BugsCEP was developed using Microsoft Office 2000 Developers Edition in the MS Windows 2000 and XP operating systems. The majority of the work was undertaken in MS Access, which provided easy access to a programming language and database motor, a combination which was not easily available when the project was initiated. The programming is entirely in Visual Basic for Applications (VBA), which is common to all MS Office software, and allows for easy communication between the applications. This enabled, for example, MS Excel to be run from within the BugsCEP code when spreadsheet functions were required, rather than having to simulate them in the database environment. The developmental potential of VBA is considerable, and it has now even been chosen as the programming language within Corel's CorelDraw and ESRI's ArcGIS. This opens up new potential routes for the future development of Bugs in the areas of diagram creation, map production and Geographical Information Systems (GIS).

2.4 Database Structural Changes

2.4.1 Bugs database structure and contents

The structure of BugsCEP is a great improvement in terms of storage efficiency and manageability over the preceding Bugs2000 structure (Figure 2.4, and see also section 2.8.3 for version history). It is these improvements that have allowed for a more flexible user interface, and more advanced searching, data-linking and querying.

The new structure was arrived at by first examining the existing data, and applying the first three normal forms (of normalization) as described by Gifford (1996) and others. This design was then expanded to include everything else that could be reasonably included. The aim was to create a structure capable of holding all the present data, plus any new that were considered important, in a logical structure that minimalized repeated information and allowed for the rigorous enforcement of validation and relationship rules. This is standard database development procedure, and can be read about in any number of manuals (e.g. Gifford, 1996) and so will not be described in more detail here. Readers interested in learning more in these areas should turn their attention to Internet searches, or the websites of any major database software developer[vii]. The final BugsCEP structure (Figure 2.3, Figure 2.5) consists of 43 main tables which contain the data, and 30 temporary tables which hold calculation results and settings. When compared to the structure of Bugs2000 (Figure 2.4), which had 19 data tables and 15 temporary tables, the increase in complexity is unambiguous.

[vii] E.g. http://www.postgresql.org/, http://msdn2.microsoft.com/en-us/sql/, http://www.mysql.com/, and others.

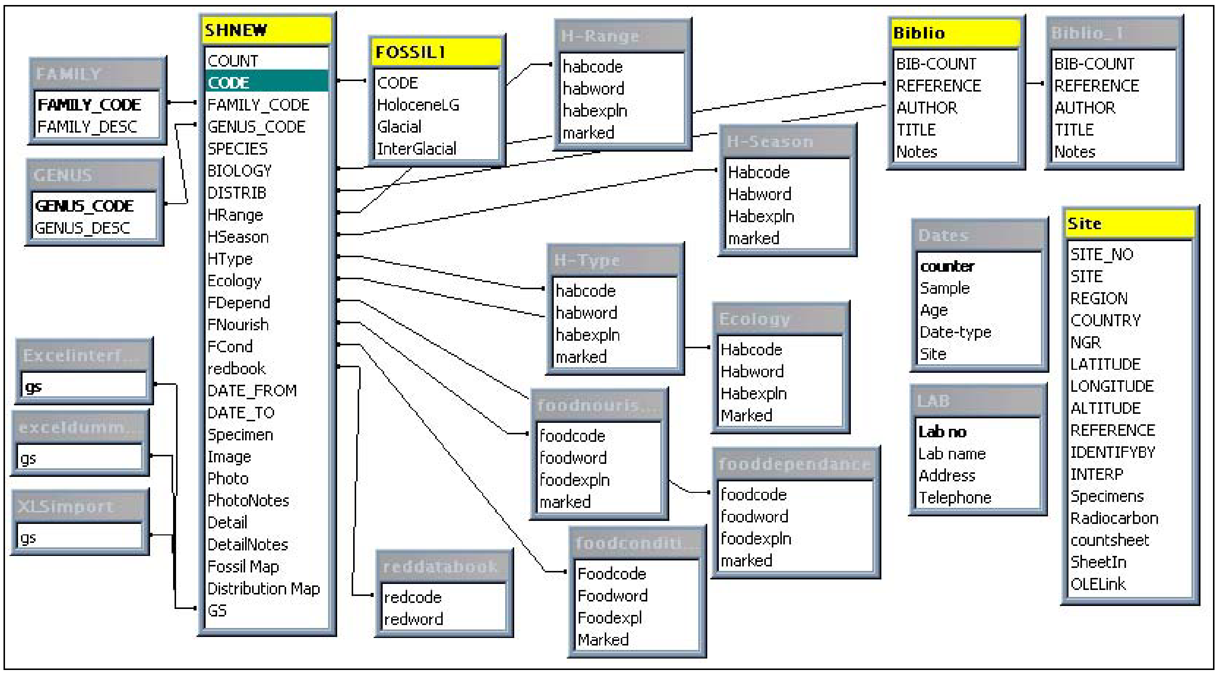


Figure 2.4. Bugs2000 database structure. Note that the relationships between the Biblio and SHNEW are symbolic, and represent programmed links rather than enforced relationships.

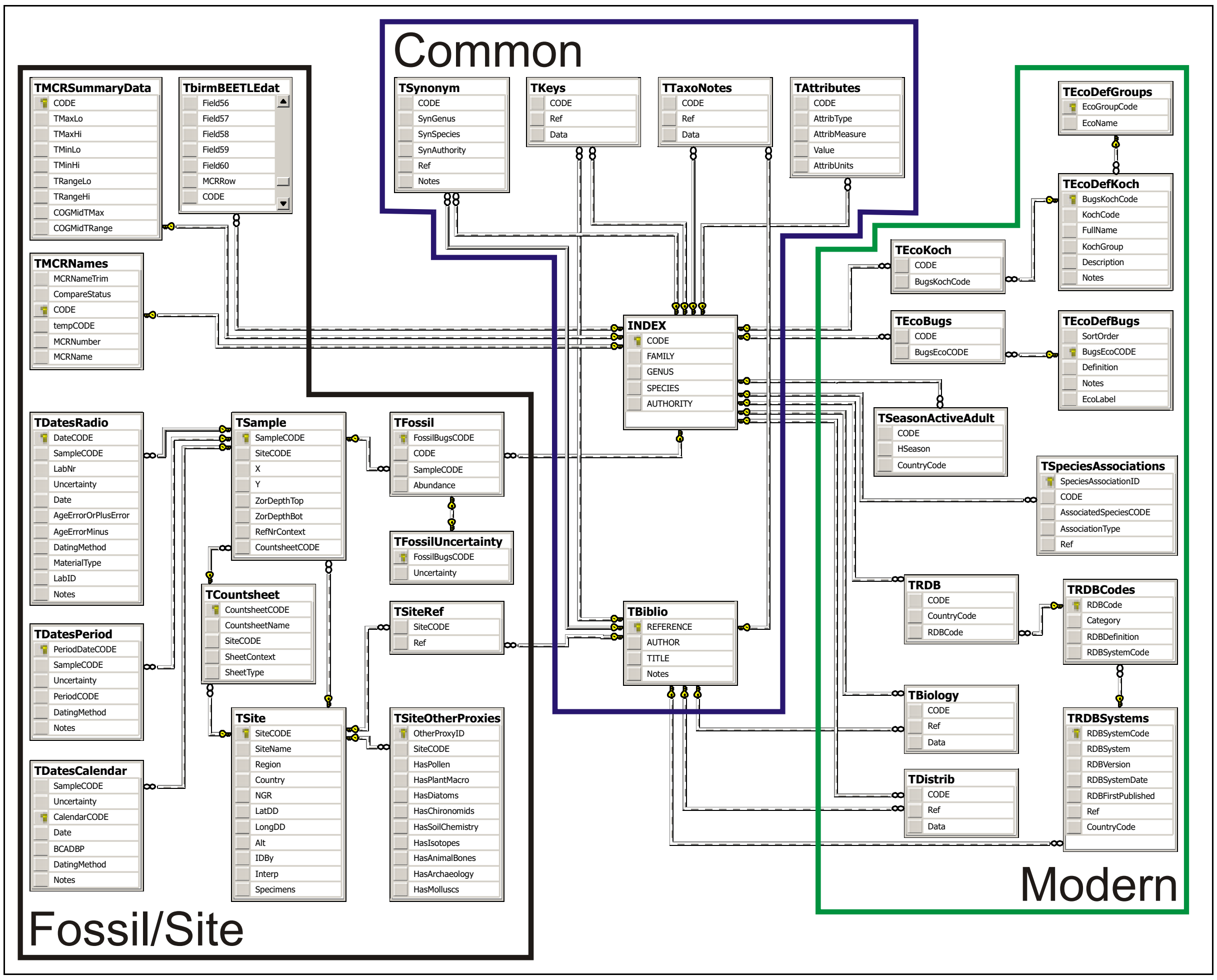


Figure 2.5. BugsCEP database structure, repeated from Figure 2.3 to allow for easier comparison with the structure of the previous version, Bugs2000 (Figure 2.4).

It was decided to drop a number of elements that were provided for in Bugs2000, in order to be able to complete the project within the scope of a PhD. Images and scanned maps were omitted, the former to be re-integrated in a later version using a more optimized external database form. Scanned distribution maps were deemed a poor, although often useful, substitute for a real-time mapping engine or GIS. Although Bugs2000 stored site coordinates there was no system built in for displaying their location on maps. The same is currently true for BugsCEP, although the author has experimented with a simple map displaying interface using VBA, and all the maps in this thesis are produced in ESRI's ArcMAP after exporting the necessary data from BugsCEP. Modern distribution maps, on the other hand, are a far more complex issue, theoretically consisting of thousands of points or areas for a single taxon just for the present day records. Recent developments in online mapping systems (e.g. GEON, Google Maps[viii]), a number of which allow the free linking of databases, may provide an efficient solution to these problems in subsequent versions of Bugs.

2.4.2 Conversion of Bugs2000 data

The conversion of data from the existing, poorly normalized, database of Bugs2000, into the almost fully normalized BugsCEP database was an enormous and time consuming task. Every item of data from the original version had to be parsed[ix], validated, and then manually checked after import. This was particularly problematic for the fossil data, which was stored entirely in notes fields in Bugs2000. After numerous attempts to extract the data, which consisted of an estimated 23 000 fossil records, themselves consisting of site name, reference, date, and occasionally other metadata, this was aborted. It was decided to recreate the data from the bottom up, converting the existing MS Excel countsheet files into the new BugsCEP format, and then importing them into new sites in the new system, parsing them into normalized data on the way. Dates were then manually added in a new Date Manager interface. A positive by-product of this endeavour was the development of the advanced file import/convert system that now forms part of the new interface. In addition, the laborious conversion attempt gave an ideal opportunity to assess the nature of the data, and devise a more optimized date handling system to cope with the wide variety of dating forms and accuracies used in Quaternary science and archaeology.

In Bugs2000, biology and distribution text abstracts were in the form of large memo field data items, where references were enclosed in curly brackets, e.g. {Buckland 2000}, and followed by the text abstracted from that source. Several of these source-data combinations would exist in each record for a taxon. A routine was programmed to extract the individual reference-data items into separate records in the new structure, with a number of validation checks to cope with missing brackets, typos and other inconsistencies. This process lead to approximately 19 300 biology records and 16 500 distribution records in the new structure, derived from the original 5 500 records in Bugs2000. The new tables can be rapidly sorted by author as well as taxon, an action that would have been impossible with the old version. The new structure also allows for the references to be linked directly to a master bibliography, from which changes can be universally applied, whereas references had to be programmatically extracted from the memo fields in the old version – a slow and cumbersome process.

2.4.3 Database structural compromises

The BugsCEP database file includes a single de-normalized master *INDEX* table, which includes repetition of family and genera names, as shown in Table 2.2. This is, technically, an inefficient way of storing information, in that the repeated text is redundant information that takes storage space, slows searches, and prevents the enforcement of referential integrity on that field. The more structurally efficient way of storing this information would be in about four tables, one for the FAMILY, one for the GENUS, one for the AUTHORITY, and an index table for the taxonomic CODE and SPECIES with foreign keys relating the other tables (as in Figure 2.1). Each of these tables

[viii] GEON: http://www.geongrid.org/; Google Maps: http://maps.google.com/maps
[ix] I.e. fields containing compounded data had to be chopped up into individual data items.

would have a unique identifier field (UID) that would allow single records for each unique data item to be linked through relationships. The normalized index might look something like Table 2.3.

Table 2.2. The de-normalized central INDEX table in BugsCEP. The headers are field names, and the rows are records.

CODE	FAMILY	GENUS	SPECIES	AUTHORITY
01.0010001	CARABIDAE	Carabidae	indet.	
01.0010020	CARABIDAE	Cicindela	sylvatica	L.
01.0010050	CARABIDAE	Cicindela	hybrida	L.
01.0010060	CARABIDAE	Cicindela	maritima	Dej.
01.0010070	CARABIDAE	Cicindela	campestris	L.
01.0010080	CARABIDAE	Cylindera	germanica	L.
01.0010122	CARABIDAE	Cicindela	sp.	
01.0010125	CARABIDAE	Cicindela	spp.	
01.0020010	CARABIDAE	Calosoma	inquisitor	(L.)
01.0020020	CARABIDAE	Calosoma	sycophanta	(L.)
01.0020050	CARABIDAE	Calosoma	reticulatum	(F.)
01.0020082	CARABIDAE	Calosoma	sp.	
01.0020085	CARABIDAE	Calosoma	spp.	

Table 2.3. Hypothetical section of a more normalized version of the BugsCEP index table. UID = Unique Identifier, the key field for this table. The AUTHORITY field could theoretically be normalized further to remove empty cells by splitting it off into a separate table.

UID	CODE	FAMILYID	GENUSID	SPECIES	AUTHORITYID
1	01.0010001	1	1	indet.	
2	01.0010020	1	2	sylvatica	1
3	01.0010050	1	2	hybrida	1
4	01.0010060	1	2	maritima	2
5	01.0010070	1	2	campestris	1
6	01.0010080	1	3	germanica	1
7	01.0010122	1	2	sp.	
8	01.0010125	1	2	spp.	
9	01.0020010	1	4	inquisitor	3
10	01.0020020	1	4	sycophanta	3
11	01.0020050	1	4	reticulatum	4
12	01.0020082	1	4	sp.	
13	01.0020085	1	4	spp.	

The reason for not normalizing the index, and in fact de-normalizing it from the previous version, is simply for the sake of the convenience of having the full species name available in one table. It simplifies query creation, and VBA coding by having the data available in this way. An alternative would be to normalize the table, and then use a saved query to create a simulacrum of the desired index, which could then be referred to in queries and VBA. In many RDBMS this would be the ideal solution, but unfortunately MS Access has a bloating problem with query use (i.e. the database file grows every time a query is run), and so this alternative has been avoided. Future versions of BugsCEP will use an alternative RDBMS, such as SQL-Server, MySQL or PostgreSQL, which do not suffer from this problem.

2.4.4 Problems and potential problems with the BugsCEP structure

Most of the problems described in this section can be attributed to the inexperience of the developers, and are most likely symptoms of learning by doing, rather than a formal education in database development. They have lead to a number complications in data manipulation or retrieval, all of which have had to be solved by small bits of extra program code.

Use of the *double data type* for taxonomic CODE and calculations

The *double data type* stores floating point numbers, and as such approximates decimal fractions with the closest binary equivalent (computers 'think' in binary). This means that some numbers simply cannot be represented by the *double data type*, and while this may generally not be a problem for the taxonomic code itself, it does cause problems where mathematics, and the *double data type,* are used to validate entered codes. Take the following case:

```
CDbl(Int(CDec(87.0290060) * 10000000) / 10000000)<87.0290060 = false
```

but

```
CDbl(Int(CDec(87.0290070) * 10000000) / 10000000)<87.0290070 = true
```

Clearly *both* formulae should evaluate to false, but they do not as the CDbl command converts the left part of the comparison to the *double data type* before comparing with the right part.

The following VBA code is used to work around the problem in the validation of new taxonomic codes:

```
Val(CDbl(Int(CDec(NewCODE) * 10000000) / 10000000)) < CDbl(NewCODE)
```

Where 'NewCODE' is the code entered by the user which is passed to the validation routine as a *double data type* variable.

See Microsoft Knowledge Base (MSKB) article 242933 for more details of this type of problem[x].

Use of taxonomic CODE as key fields

A standard rule of database development is that business data and structural data should not be mixed. Business data is that which has meaning, whereas structural data is that which primarily provides the anchors for the relationships between tables and allows for the efficient use of lookup tables. Taxonomic CODE is clearly business data, and its use as a primary key in the *INDEX* table is clearly a structural function (see Figure 2.3).

Whilst this normally would not cause any operational problems, and in fact makes debugging easier in some respects (as the taxonomic CODE is visible in the majority of tables), it does make the implementation of major changes to the nature of taxonomic codes awkward. For example, the choice of data type used for the taxonomic code field was unwise, as explained above, and it should really be changed. Due to the structural nature of the CODE field, however, this would involve opening the design view of each and every table where the field is present and changing the data type. It is not, however, possible to change the data type of a field that is part of a relationship, so each and every relationship would first have to be deleted. These would then have to be recreated subsequently. These tasks in themselves would only take a number of hours, but then the program code, which has routines created specifically for handling taxonomic codes in the double data type, would have to be checked and adjusted to cope with the new data type. This task could take days, and would then have to be followed up with a comprehensive testing of every part of the program. Changes of such a

[x] http://support.microsoft.com/kb/242933/en-us

comprehensive nature also usually lead to unforeseen consequences, and would undoubtedly need the addition of new error handling routines. Had an alternative unique identifier been used as the key field instead (cf. the TaxonID field in the *tblTaxaMasterInsects* table, Figure 2.1), then the data type of the separate taxonomic CODE field (cf. the TaxonomicCode field in the *tblTaxaMasterInsects* table, Figure 2.1) could have been changed with considerably less effort. This structure would also allow for the easier implementation of multiple taxonomic systems.

Use of a decimal separator in taxonomic codes

The European Taxonomic Code uses a decimal separator to divide between family and genus numerical equivalents (see section 3.1.1.1). Numbers are, logically, best treated as numbers in databases and program code, and so the taxonomic code is stored and manipulated as such[xi]. The English language uses a decimal point ('.') as a separator, whereas a number of other languages, including Swedish, use a comma (','). Structured Query Language (SQL) can *only* handle a point as a decimal separator[xii], and will produce an error if a comma is used. On non-English systems, taxonomic codes are retrieved with a comma instead of a decimal point, and are thus passed to SQL statements with the former, causing a syntax error. The following code therefore is used to replace the comma with a decimal point in codes before they are inserted into SQL statements:

```
CommaPos = InStr(1, CODEAsString, ",")

If CommaPos > 0 Then

     CODEAsString = Left(CODEAsString, CommaPos - 1) & "." &
     _Right(CODEAsString, Len(CODEAsString) - CommaPos)

End If
```

A related problem is that SQL cannot naturally cope with the use of apostrophe's (′) in string values. These must be doubled up in order to not cause an error on processing the SQL.

The following function is used in BugsCEP to cope with any value which includes apostrophes:

```
Public Function SQLEncode(sqlValue As String) As String

    SQLEncode = CStr(Replace(sqlValue, "'", "''"))

End Function
```

It returns a version of the passed string 'sqlValue' which can be inserted into SQL. For example:

```
SQLEncode("test's") = test''s
```

Use of a limited number of prefix zeros in unique identifiers

Unique identifiers are automatically generated in BugsCEP for all new data items according to a template consisting of a four letter prefix, which designates the type of data, and six numbers, which are increased in order of creation. The six digit number allows a maximum total of 1 000 000 data items in a particular type. This limit seemed more than sufficient at the time of creation, and was in fact one digit, or a factor of ten more than the calculated need. With the current c. 88 953 fossil record data items this limit should theoretically be adequate for at least the near future, considering the small size of the palaeoentomological community. Unfortunately, deleted identifiers are not reused unless they were the last ones created, so due to the revision of lists and a number of experiments the highest fossil record identifier is at the time of writing 'FOSS098663'. This suggests that the number part of the actual identifier codes will usually be about 10 % higher than the physical number of records, and consequently that the probable limit to the number of records will be about 909 000.

[xi] Note that other unique identifiers in BugsCEP are either stored as alphanumerics or integer numbers to avoid the problems described here, among others. Of course, these are not without their problems as well.
[xii] At least in the MS Access implementation.

In the event of upsizing of the database to a more powerful management system, such as SQL-Server or MySQL for example, and assuming a future with prolific palaeoentomology work, it would probably be a good idea to revise this system to avoid limits being reached. There are wide ranging programmatic implications of changing the identifier code system, in terms of code generation and validation, which would take time to resolve, but none are too complex to be a critical hinder. The use of simple, integer identifiers would reduce the potential for future problems.

2.5 Program Development

2.5.1 Summary of deficiencies in the previous version (Bugs2000)

Bugs2000 had a significant amount of room for improvement. Below are listed some of the main limits of the system, from the point of view of the user, although some are essentially structural in nature.

- Synonyms not searchable
- Only one countsheet per site
- Only one reference per site
- Limited number of search terms enterable (three biology + three distribution + three Red Data Book)
- No climate reconstruction (MCR)
- No built in system for quantitatively summarizing faunas, or calculate other sample based analyses
- Fossil record is memo/notes field based, and not searchable
- No facility for storing samples, and associated metadata
- No facility for handling dates linked to samples
- No ability to import imperfect data – i.e. data must be in exactly the right format, and only include the taxa names found in Bugs2000
- Inflexible Red Data Book (RDB) system that only allows UK rarity data to be entered
- Koch EcoCodes interface cumbersome, and based around bad structure
- Taxonomic notes mixed with biology data
- Limited reporting facilities

A number of these limitations have been identified by user feedback, and all of the above have been resolved in BugsCEP.

2.5.2 Main improvements in the BugsCEP system

What follows is a brief annotated list of the main improvements of the BugsCEP program interface and functions over previous versions. Chapter 3 includes detailed descriptions and screenshots of the actual interface parts.

The Main BugsCEP screen

Effective use of tabs to reduce clutter, whilst increasing the amount of information easily available for a species. Synonyms and taxonomic notes are now separated from the main biology data for ease of access. Alternative navigational facilities, including a searchable synonym browser and code browser are provided.

Users with administrator rights can easily switch to 'Edit Mode' and make changes or additions using the same tabs, but with editor user friendly functions enabled. A password is required for administrator access, and is available from the author on request. This system reduces the risk of accidental data loss by those not familiar with the program.

Site Management interface

Previous versions had no way of getting a rapid overview of all sites stored in the database, the Site Manager now provides this. It displays summary data for each site on selection from a scrollable list, along with buttons to quickly access the site's description (metadata), countsheets, references and dating evidence.

Multi-level nesting of data

A site may now hold an unlimited number of countsheets, enabling complex sites to be sub-divided into manageable units, or multiple years of pitfall trapping expeditions to be stored logically under the same site name. A site may now have any number of references attached to it, which is necessary when sites are re-examined, or analysed over a long period of time and resulting in several publications. Unfortunately references cannot be attached to a countsheet, but the latter can be named to reflect their source.

Any number of dates may be ascribed to samples, which themselves are always linked to a particular countsheet. This allows for re-dating information to be stored, as well as a mix of radiometric, calendar (or archaeological) and general period dates to be ascribed to the same sample.

Search system

A new, multi-stage search system has been developed, that allows searching on the majority of data fields, including both by keyword and item selection. Search stages are logged, and the latest step can be undone. Search logs are exported with reports, and the reporting facilities have been extended. Search results can also be exported as a Bugs Countsheet, for re-import into a site. Searches are saved between sessions, so that they can be continued even after restarting the program.

A powerful new function to report the sites where search result species occur has been added.

Internal Site/Countsheet Management

Although Bugs2000 could be used to create species lists, these had to be opened in MS Excel to add samples and abundance data, which was a troublesome and error inviting task. BugsCEP allows for all countsheet management tasks to be performed within the program, including the creation of species lists, sample lists, specification of metadata, and addition of abundance data. It has a number of built in validation routines to prevent the missing of taxa or samples, or the entry of illegal abundance data. This bypasses many problems caused by the free range given to users when entering countsheets data in MS Excel. In the old version, it was difficult to replace a taxon, and one risked losing abundance data if not very careful. BugsCEP provides a button for this.

The new system has been enabled by the normalization of the fossil record system, which now includes a record for each and every presence of a taxon in a sample.

MCR – Climate reconstruction

The MCR (Mutual Climatic Range) thermal reconstruction method of Atkinson *et al.* (1986) has been implemented in BugsCEP, and can be run on any site that is stored in the database. In addition, output of thermal envelopes and samples species lists is provided, as well as a temperature based species presence prediction system. The latter can be used to help answer questions on the potential effects of climate change on the insect fauna.

More advanced functions, including jackknifing, to calculate the internal reliability of a reconstruction, and relative cold/warm component identification are under development.

BugStats – Environmental reconstruction

Using the new Bugs EcoCode habitat classifications, users can summarize the habitats represented in samples numerically and graphically. A total of 16 possible calculation option combinations are available, including exclusion of generic level identifications, log transformation, abundance weighting and standardization by sum of represented environments. This is potentially a very powerful tool in palaeoenvironmental reconstruction, and provides flexible output in MS Excel format. A report can be produced listing the habitats represented by the species at a site on a sample by sample basis, with abundance values.

In addition, a correlation coefficient calculation system was created, which can output an MS Excel file containing a matrix showing the measure of similarity/difference between each and every sample at a site. Currently only the Bray-Curtis modified Sørensen's coefficient of similarity (Southwood, 1978) is activated, but more coefficients will be added at a later date.

Advanced file importer

The file import system accessed through the countsheet manager includes automatic synonym replacement, including a user customizable record of manual replacements. It also includes an easy to use interface to allow for the resolving of name differences due to typing and spelling mistakes. Converted files are imported directly as countsheets, removing the need for repeated export and import. The files can be exported in the BugsCEP format at any time, once imported.

Multinational and international Red Data Book (RDB) rarity recording system

Records of species rarity (see section 3.1.2.3) are no longer limited to the UK RDB, data can be entered for any number of countries, and for international records. This makes BugsCEP potentially much more useful for studies on biodiversity changes and for monitoring endangered species.

Enhanced reports, and more of them

The standard reports from Bugs2000 have been dramatically improved, now including full references and site summary data, as well as the biology and distribution data for all species found at a site. A number of additional reports have been added, including more flexible export to either MS Word (as an RTF file) or MS Excel. In particular, users can select from a number of variants of a standard report with different levels of information. This facility is available for search results and single species, and will be implemented and expanded elsewhere in the program.

Bibliographies for sites or species can now be exported in report form to either MS Word or Excel.

2.6 Testing

2.6.1 Developer testing

Until 2005, Bugs2000 and BugsCEP, and all components, were tested on MS Windows 98, 2000/NT and XP machines. Windows 98 was dropped at that point due to its increasing scarcity, but as Windows 2000/NT is still considered a stable operating system in many laboratory environments it is still supported. Due to lack of backward compatibility, a common problem with Microsoft developer environments, the setup files for installation on Windows 2000 must be packaged on a Windows 2000 machine and not a system running Windows XP.

2.6.2 User-based testing

The user test base has primarily been kept small due to two reasons: 1) managing feedback from users during testing of unfinished software consumes a lot of time; 2) there is a tendency for test users to start actually using the unfinished features, which is not advisable. Undergraduate students have been used as in house testers at the Environmental Archaeology Lab, and have provided considerable help in particular with the BugsMCR component.

User testing has been particularly useful in aiding the development of a more intuitive interface, as the developer's technical understanding of the system can often give him a false impression of simplicity in interfaces. That is to say, after having programmed the system for more than five years the developer's understanding of simplicity will be different from anybody else's. In particular, test user feedback has lead to:

- additional explanation of the frontend-backend linking procedure on first run
- help files being included in this release (they were planned to come later)
- the drop down site choosers in BugStats and BugsMCR
- fine tuning of BugStats and BugsMCR output files and reports
- numerous layout aspects, including the use of numbered stages in complex task sequences

2.7 Presentations and Publications

Although there is limited scope for the production of individual papers within a monograph orientated PhD, it has been possible to produce a number of publications. This has been deemed necessary, in that papers are the accepted scientific discussion medium amongst the older part of the scientific community[xiii], and that software essentially needs supporting publications to be noticed. Conference presentations have been targeted over journals, in that these presentation and discussion fora provide a greater level, and often more immediate, feedback, and allow demonstration and distribution of the software to interested parties.

2.7.1 Publications about or with direct use of Bugs

Buckland, P.I. & Buckland, P.C. (2002). "How can a database full of Bugs help reconstruct the climate?"

Provides an overview of the BugsCEP development strategy, with an emphasis on the MCR and related data acquisition aspects. Provides a blueprint for a GIS based system for improving the MCR dataset.

Buckland, P.C. [Assisted by Buckland, P.I. & Hughes, D.] (2005). "Palaeoecological evidence for the Vera hypothesis?"

A practical application of the BugStats environmental reconstruction component of BugsCEP, showing a variety of sites useful in elucidating the Vera hypothesis (Vera, 2000) with respect to the British Isles. Summary site lists were produced for this report using custom build queries in the BugsCEP database, and time slice maps then created using ESRI's ArcView.

[xiii] This is almost certainly less true for the Google-Wiki generation.

Buckland, P.I., Buckland, P.C., Panagiotakopulu, E. & Sadler, J.P. (in press). "A Database for Egyptian Entomology"

An overview of the Egyptian spin-off of Bugs2000 (see section 2.8.4.2), explaining its structure and use.

2.7.2 Unpublished presentations

2002 (1 May) "The Royal Entomological Society. Entomological Computing And Technology Special Interest Group – Recent Advances In Entomological Computing and Technology" London, UK.
Presentation: Buckland, P.I. & Buckland, P.C. "BUGS in the system: a database of Quaternary entomology and Holocene habitats (with a footnote on EGBUGS, an Egyptian version.)".

2.7.3 Poster presentations

2001 (19-22 August) "VIII Nordic Conference on the Application of Scientific Methods In Archaeology" Umeå, Sweden.
Buckland, P.I., Buckland, P.C., Engelmark, R.E., Sadler, J. & Atkinson T. *"Climatic reconstruction through beetle proxy temperature data"*. (NB. This poster is the same as the PAGES poster below).

2001 (27-31 August) "PAGES-PEPIII -- ESF-HOLIVAR Past Climate Variability Through Europe and Africa An International Conference". Centre des Congrès, Aix-en-Provence, France.
Buckland, P.I., Buckland, P.C., Engelmark, R.E., Sadler, J. & Atkinson T. *"Climatic reconstruction through beetle proxy temperature data."*

2003 (23 June - 4 July) "Quantitative climate reconstruction and data-model comparisons" Environmental Change Research Centre, UCL, UK.
Buckland, P.I. *"Climate & Environmental reconstruction from fossil beetles"*.

2005 (13-14 November) "Joint HITE-POLLANDCAL Conference: Human impact on terrestrial ecosystems on long to short term scales with an emphasis on pollen calibration and quantitative reconstruction of past land-cover changes". Umeå, Sweden.
Buckland, P.I. & Olsson, F. *"Using insect analysis and ecological coding to reconstruct human impact and environmental changes during the late Holocene"*.

2.7.4 A note on other publications using Bugs (2000/CEP)

Although Bugs2000 is quite widely used in the palaeoentomological community it is very rarely cited, with the exception of the developers and their immediate colleagues. This is not an uncommon situation for databases, and it appears that authors are not always aware that they should cite database software, and perhaps regard it simply as a passive tools in their work. The latter is far from the truth, as the use of Bugs, as opposed to literature or other data sources, directly influences the interpretations made by way of the extent of its data and tools. Citing the correct data source and version is essential for the reproducibility of scientific results, as much for fossil beetle work as for any field. For example, conclusions drawn from DNA analyses using a demographic database on the population of Iceland could be different to those drawn from a database on the UK population, the importance of citing the correct data source being obvious in this example. Computer software is afforded the same copyright status as scientific papers, and it is thus not only ethical to cite their use but also a legal requirement. Databases also have the same status as any archive, and must be cited as a data source when used. Unfortunately the citation indices are yet to cater to data sources, and it is currently not possible to track the degree of citation of the Bugs software. The publication of this thesis may help the situation to a degree, but the authors recommend using the citation form suggested in the software interface, which always reports the versions of the program components and database being used.

2.8 A Brief Developmental History of Bugs

The Bugs project essentially began c. 1989 with the development of the first Bugs database (Sadler *et al.,* 1992). The software was primarily designed to reduce the amount of time required to look up ecology and habitat data for beetle species found in samples.

Previous versions of Bugs have been described elsewhere (Buckland, 2000), and they are summarised below along with their primary references. The intention here is to give an overview of the long term nature of the semi-professional development of a research tool, and to illustrate the tremendous amount of work that has gone into the Bugs project. The author of this thesis has been involved in the project since the early 1990s, and took over the software development side in 1996.

2.8.1 Design criteria for Bugs

The following criteria formed part of the Bugs initiative from the very beginning.

1. It should run on standard home PC, either standalone or through commonly available software.
2. It should be easy to distribute, and independent of installed database software.
3. It should be user friendly and aimed at a poorly computer literate user base.

The following criteria were arrived at during the development of BugsCEP and its components, and only became of real importance in the latest version:

4. It should be expandable and updatable – in terms of both the data and the program.
5. All calibration data should be visible and methods transparent.

2.8.2 Implementing the design criteria

Although the design criteria may appear relatively simple, their implementation has not always been straight forward.

1. Bugs has grown enormously since inception – the original was only about 8 MB, and BugsCEP is between 50 MB and 150 MB depending on installation. Storage space has at various stages been a concern, but technology has fortunately progressed faster than the development of the database and this has ceased to be a major concern. The amount of data in BugsCEP is no longer something that would be considered particularly large, and most current home PC's can run BugsCEP without significant delays when processing.
2. Although BugsCEP is dependent on the MS Access database management system, the developer software includes that facility to create runtime packages for distribution. That is to say that MS Office Developer Edition includes the tools necessary to produce database orientated software that can be distributed to computers that do not have the Access database software installed.
3. This was relatively difficult with the original MS-DOS version of Bugs due to the inflexibility of the dbase4 – Clipper environment (Sadler *et al.,* 1992). With the release of MS Windows, creating GUIs (Graphical User Interfaces) became increasingly easy, and subsequent releases of developer environments such as MS Office Developers Edition have only added to this. A major disadvantage of the MS-DOS version of Bugs was the need for the user to enter search criteria as exact SQL 'where' statements. Although this allowed for accurate retrieval it was not ideal in that the majority of users were unfamiliar with SQL, and even those that were had trouble remembering where to put brackets and commas. Later versions replaced this system with a more intuitive interface where users could enter search terms for biology, distribution and RDB, and select the appropriate search logic from drop down boxes. BugsCEP takes this a step further by providing a comprehensive search system covering the majority of data areas. Much of the usability of recent versions of Bugs is

thanks to the development of object orientated programming, and the relative ease of creating user interfaces in the MS Windows environment, through developer tools which have implemented this.

4. Updates have always been provided for download when significant changes to the data or program have been implemented. Data updates have proved problematic only where users have entered a large amount of data themselves. Unfortunately the system does not include the ability to update an existing copy of the data or program on a user's computer, only replace it. This means that any additions users have made will be overwritten when the update is installed. The separation of Bugs into frontend and backend (program and data) files reduces the problem to great extent by allowing the program part to be updated without overwriting the user's data. The data update problem still remains to be solved however, and at the moment it can only be advised that users export their sites before updating. Bugs was not designed with the end user addition of species data in mind, although it is understood that users with particular specialisms could find this necessary. The current strategy is to keep species and bibliographic data entry centralized with the developers. Should BugsCEP become widely used, this strategy may have to be reviewed with respect to a clearing house or remote web based data entry facility, although such options would require considerably more management than the current system.
5. This criterion was arrived at due to an awareness of the increasing mathematical complexity, and reducing transparency of palaeoenvironmental reconstruction methods. The comment by a conference poster presenter, whose results had proved to be statistically insignificant: '...we hope that more advanced statistical methods will find a significant relationship' suitably summarising the trend. An acceptance that more complex methods will provide better results is a little naive, and at odds with Occam's metaphorical razor. In addition, the assumption that smaller error estimates always mean a better reconstruction is erroneous, and the complexity of the data manipulation often hides many unquantified errors. It was thus the intent of the author to keep any statistical manipulations provided by BugsCEP simple and effective. Beetle abundance data are subject to many taphonomic problems before the final dataset is arrived at. These problems are difficult, if not impossible to quantify (see Chapter 4) – and thus have errors that are difficult or impossible to assess. More complex statistics applied to these datasets often complicate these errors even further.
   BugsMCR fails very slightly in fulfilling this aim in that it is currently not possible to directly examine the thermal envelope of a species within the program, although the user can create samples with single species and examine the envelopes in MS Excel. This is an unfortunate consequence of prioritization of features, and will be included in a future version.

2.8.3 Version history

Bugs

Primary reference: Sadler *et al.* (1992)

The original program was developed by Jon Sadler as part of his PhD at Sheffield University, UK, with assistance from Mike Rains and Paul Buckland. It was constructed in dBase and compiled with Clipper, and ran in MS-DOS. A manual was produced by Philip Buckland in the early 1990s.

Bugs v.2

Unpublished

Version 2 was developed in VisualBASIC as a standalone MS Windows application, but dropped in the early prototype stages due to lack of time and funding. This was the first Windows version of Bugs.

FoxBugs

Unpublished Masters Thesis, Sheffield University, UK

FoxBugs was developed in the Borland FoxPro DBMS as part of an MSc project in computer science. A prototype system was completed, but further development was dropped when Sheffield University withdrew support for FoxPro, and switched to Microsoft products. Ironically enough Microsoft later acquired FoxPro.

Bugs for Access 2

Primary references: Yuan Zhou (*Unpubl.*); Buckland *et al.* (1997)

The first incarnation of the present series of Bugs programs developed in MS Access. The majority of programming was undertaken by Yuan Zhuo Don, with assistance and later refinements from Philip Buckland. Yuan Zhou wrote Bugs as part of his Computer Science MSc (1995) at Sheffield University, UK, and stayed with the project until 1996.

Bugs for Access 97

Unpublished (Philip Buckland)

The Bugs2000 interface began to take shape in the MS Access 97 RDBMS (which superseded Access 95 so quickly that few people noticed), with a number of improvements in interface design over the Access 2 version.

Bugs2000

Primary reference: Buckland (2000), website http://www.bugs2000.org

Bugs2000 was developed in MS Access 2000 and Visual Basic for Applications (VBA), and went through several versions leading up to the most recent v.5. Most of the releases prior to v.5 were bug fixes of various forms, the most common problems arising from variations in MS Windows installations and different versions of Windows. Data were added continually by Paul Buckland, and programming was initially by the author of this thesis, around some of Yuan Zhuo's original core, converted from Access 2. Additional maintenance and restructuring work was carried out by a group of MSc students under the umbrella of Sheffield University's 'Genesys Solutions'[xiv] software development house, as part of their joint MSc computer science project. The Genesys team undertook the following work:

- split the database into frontend-backend, including facilities for relocation of the backend
- a degree of structural normalisation (some of which was subsequently reversed in BugsCEP, see section 2.4.3)
- fixed a number of errors including those associated with non-English characters in references
- reconstructed the countsheet system to work from externally stored MS Excel files
- implemented a degree of user level security

BugsCEP – Bugs Coleopteran Ecology Package

Primary reference: This thesis, website http://www.bugscep.com/

BugsCEP is the version described in this thesis, developed in the MS Office 2000 Developer environment, using MS Access and VBA, in addition to MS Excel for a number of graphing routines and export functions. BugsCEP is a completely new piece of software built around an entirely new,

[xiv] http://www.genesys.shef.ac.uk/

more efficient database structure, as described below. The database design, programming and conversion of data from the previous version was undertaken by the author of this thesis. Paul Buckland has been responsible for the majority of the data entry.

2.8.4 Bugs spin-offs

The concept of Bugs inevitably sparked interest from researchers in other branches of Quaternary science, and the development of version for other proxies were considered.

2.8.4.1 Slugs – Molluscan Database

Primary reference: Unpublished, but mentioned in articles as given below.

A prototype molluscan database was created from the Bugs (v.1. MS-DOS) (Sadler *et al.,* 1992) and subsequently Bugs97 systems (v.2, Figure 2.6), and a small amount of data entered to demonstrate the possibilities. Slugs remains suspended and unsupported, but is available should anyone be interested in taking up development.

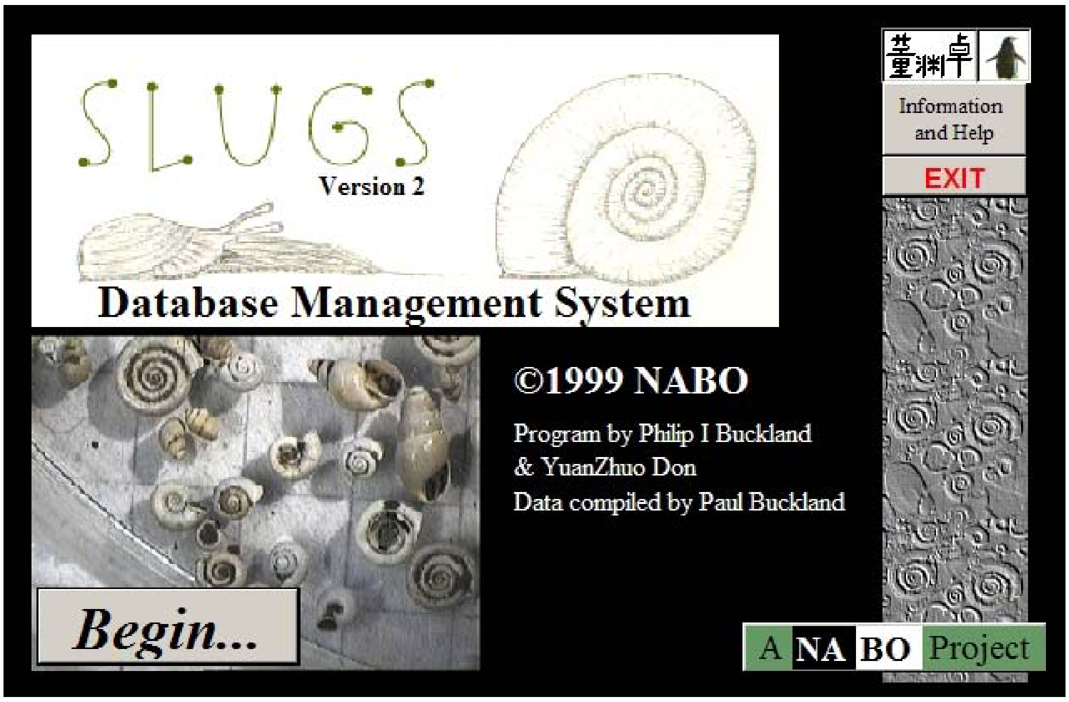

SLUGS
Version 2
Database Management System
Information and Help
EXIT
©1999 NABO
Program by Philip I Buckland
& YuanZhuo Don
Data compiled by Paul Buckland
Begin...
A NABO Project

Figure 2.6. Start-up screen of Slugs v.2 prototype molluscan database

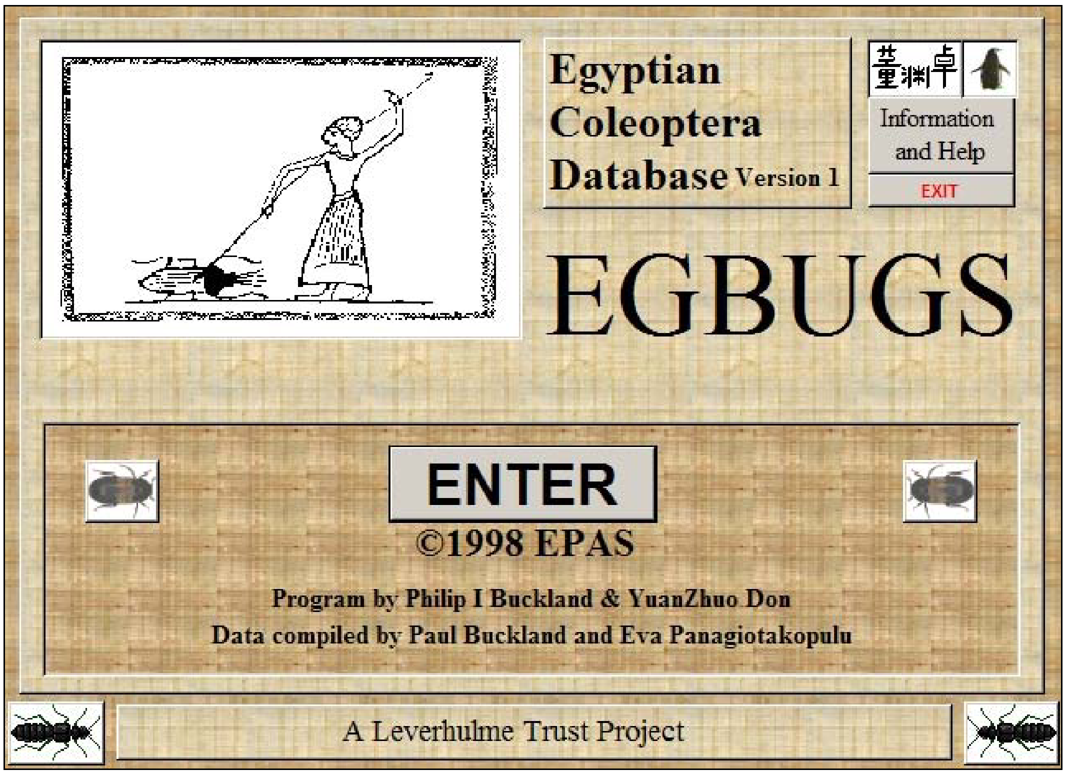
2.8.4.2 EgBugs – The Egyptian Coleoptera Database

Primary reference: Buckland *et al.* (*in press*)

EgBugs (Figure 2.7) was developed from the Bugs97 system with an aim towards providing a research and teaching tool for use in the North African and Eastern Mediterranean regions. Its emphasis was on species of use in archaeological investigations in these regions, with the initial scope centred around Egyptian archaeology (hence the name). Programming was by the author of this thesis and Yuan Zhuo Don, and data entry, including the adaptation of the Central European taxonomic code to Egyptian species, was by Paul Buckland and Eva Panagiotakopulu.

The environmental and geographical diversity of beetles makes them ideal for studying environmental change, but also means that attempting to make databases which encompass larger geographical scope will encounter problems. As the database is built around a standard series of codes developed by Koch

(1989) for the Central European beetle fauna, there are problems when the system is applied to other areas, and an international standardization of taxonomic codes would help to resolve the problem of uniquely identifying each taxon numerically. In addition, there is the problem that the habitat preference of species may be geographically dependent. *Sitophilus granarius* L., the grain weevil, for example, is entirely synanthropic in northern Europe, where it only survives in habitats created by humans, whereas it is thought that the species' natural habitat in the Middle East may be in nests of seed storing rodents (Buckland, 1990). A database orientated system, which utilized codes for species ecology encompassing both these regions would have to have functions built in to enable users to limit the scope of fossil habitat related investigations by region. This would avoid the problem of projecting the natural Middle Eastern habitat onto the prehistory of Northern Europe, where it is infeasible. An alternative, as chosen in the current scope of the Bugs project, is to limit the geographical extent of the database to a region where assumptions of habitat specificity are reasonably certain. There is inevitably an element of circular reasoning in doing this, and where long timescales are involved the danger of excluding possible interpretations on the grounds of geographical improbability does present potential problems. An understanding of Coleopteran ecology and biogeography, along with the fundamentals of palaeoecology and Quaternary geology are thus essential when interpreting the results of any fossil insect fauna.



Egyptian Coleoptera Database Version 1

EGBUGS

©1998 EPAS

Program by Philip I Buckland & YuanZhuo Don

Data compiled by Paul Buckland and Eva Panagiotakopulu

A Leverhulme Trust Project

Figure 2.7. Start-up screen of EgBugs – the Egyptian Entomology Database.

2.8.4.3 *Other embryonic versions*

A plant macrofossil version was investigated, but dropped due to lack of momentum and the need for more structural changes to incorporate ethnographic data and plant use information, along with other modern reference data of interest to archaeologists. A preliminary database structure has been sketched by the author of this thesis, Roger Engelmark and Karin Viklund at the Environmental Archaeology Lab in Umeå, Sweden, and forms part of the SEAD project (see section 1.2.4) prototype multi-proxy database for environmental archaeology (Buckland *et al.,* 2006).

2.8.5 Other Related Software

2.8.5.1 MS-DOS MCR Software – RECON & RECON2 and related programs

Primary references: Atkinson *et al.* (1986), Perry (1986).

Although the original MS-DOS based Mutual Climatic Range software was never formally published or distributed[xv], it constituted a major advance in the computerization of palaeoentomology. The two versions, commonly referred to as MCRBirm (Birmingham) and MCR UEA (University of East Anglia) after their respective origins, provided computerized climate reconstructions from beetle assemblages. The reconstruction programs were called RECON and RECON2 respectively, the Birmingham system being the first to be developed. The packages also included a number of tools for data compilation, calculation and display, some of which were restricted to mainframe terminals. Despite the limited user-friendliness of the MS-DOS environment, this was a significant improvement over the use of overlain transparency sheets for climate reconstruction. It was not only faster, but more accurate in terms of removing a number of significant human error factors from the calculation process. Both reconstruction programs ran in MS-DOS, and were compiled FORTRAN programs. The author of this thesis has not been able to run RECON, due to compatibility issues, but was able to extract the thermal envelope data for use in BugsCEP as described below.

2.8.5.2 BugsMCR (standalone version)

Primary references: This thesis, and design outlined in Buckland & Buckland (2002)

BugsMCR is the name of the Bugs and Windows implementation of the MCR method (Atkinson *et al.,* 1986), and was used as the name of the standalone prototype software which is now integrated in BugsCEP. It went through several incarnations before the final prototype was arrived at, and was made available for download from the Bugs2000 website.

The thermal envelope data from the MCRBirm (RECON) was converted and imported into an MS Access 2000 database, and forms the backbone of BugsMCR. Intuitive graphical user interfaces, and export and graphing functions provided a significantly improvement in usability over the previous MS-DOS MCR implementations. A number of more advanced calculation tools were developed within the program, including an application of jackknifing, and rudimentary temperature based prediction (included in BugsCEP). The program was successfully used in teaching in Umeå (Sweden), Bournemouth (UK) and Royal Holloway (UK), and was tested in a number of other departments.

2.8.5.3 BugStats (standalone version)

Primary references: This thesis, and used in producing results for Buckland (2005)

BugStats (previously spelt BugsStats) is the name of the environmental reconstruction component of BugsCEP, and was also the name of the standalone prototype software. BugStats provides semi-quantitative graphical habitat reconstructions from fossil assemblages (although modern studies may also have use for it). It is particularly useful for illustrating changes in habitats over time, and was built primarily for helping in the interpretation of Quaternary sequences. A number of calculation and output options have been built into the current version, including correlation coefficients.

[xv] The PC software was available for free on request.