Messy data – clean software – brilliant results?

Candlin, Francis; Morgan, Nicholas J.

Veröffentlichungsversion / Published Version
Zeitschriftenartikel / journal article

Historical Social Research, Vol. 15 — 1990 — No. 1, 72-78

Messy Data - Clean Software - Brilliant Results?

Francis Candlin and Nicholas Morgan*

The historian's attitude to data and the way he uses it presents something of a challenge to the software writer and computer user. The historian's data is to be found in crumbling, smelly and dusty ledgers, buried in an obscure part of a crumbling, smelly and dusty archive. Opening one of these ledgers, he discovers that the entries are scrawled higgledy-piggledy over the page. The ink is faded: the language, if not if not in an obscure Latin, will consist of original clerk's shorthand and terminology, the meaning of which was lost centuries ago. Our job is to translate all of this into authorative conclusions, gleaming graphics and credible generalisations.

Before discussing potential ways of making the impossible merely difficult, it is worth reminding ourselves of why historians use the computer to produce results from this kind of data. Databases of historical records can act as a reference tool - in other words we can use them to extract information on named individuals or houses or other items of interest. However, it is unlikely that many historians would be prepared to devote massive amounts of time to building a large database on the offchance that it contains references to specific individuals. Generally historians use the computer because they wish firstly to have the ability to pursue the particular, secondly to have the ability to identify groups according to particular shared characteristics, and thirdly in order to treat data relating to groups in some quantitative way. The historian wants to discover the make up of the early family, and to do this he loads a number of parish registers onto the computer. He asks questions of he data such as the number in each household, perhaps comparing one area with another, one social class with another or one period with another. Alternatively he might be interested in patterns of trade, and so he loads shipping registers onto the computer and performs simple calculations on each of the relevant commodities. At the same time, in both cases, the original material is stored on the computer in a way that serves the enquirer searching for a particular early modern family, or a particular ship or commodity.

These examples are used to emphasise the fact that historians frequently use the computer to store, retrieve or sort textual information which they wish to describe in a numeric way. The calculations may be simple (pos-

* Address all communications to Francis Candlin, D.I.S.H. Project, University of Glasgow, Laboratory, 2 University Gardens, Glasgow G12 8QQ, Great Britain.

sibly consisting of no more than a frequency count on different items of interest) but nonetheless scholars are forcing the computer to treat data in ways it was not originally intended to be used. It is probable that the data was collected purely to serve as a record of a transaction or other happening. It goes without saying that if the historian does not properly understand why and how the record originated he should not use it.

One way of making historical data suitable for computer processing is to code everything before it goes in. This method was much used in the past (particularly by self-styled 'social science historians'), if only because the expense of the hardware made it otherwise impossible to produce reasonable quantities of machine readable data (1). This is a method still pursued in a thinly disguised form by many who are trapped in the technology of the late 1960s and early 1970s. Leaving aside discussion of the not-too-user-friendly aspect of dealing entirely with strings of digits, a heavily coded source could, ironically, be less effective for general analysis than an uncoded one. When the historian codes data, he imposes his own interpretation on it -this is a perfectly valid operation for any particular research undertaking. Coding may not be so valid, however, when a second piece of research takes a completely different angle of analysis on the data. Problems with coding become acute when the database is passed onto another person who may not agree with the coding scheme used. In any case, if the other person does not have access to the raw data of the original source he is unable to verify if the codings are relevant to his particular purpose. Coding of course assists generalisation but it destroys the particular that makes up the lifeblood of both historical narrative and analysis (2).

All this boils down to a requirement that the working data should be as close to the original source both in content and in format as possible. Obtaining effective results becomes much easier if historians allow themselves to produce made to measure programs for each database they wish to analyse. There is nothing wrong with this approach - indeed, in some cases the data that we are looking at is so complex no other method is possible - but the obvious demands on time and finance would be beyond the constraints of most research groups or history departments. The DISH Project in Glasgow, constrained by lack of resources, is committed to reusing software across a range of research areas in a common and easy-to use computing environment (3). This means that each program has to be able to cover a large number of operations with as few tools as is possible. The software also has to be easy to use. Knowing that other institutions face the same constraints as Glasgow DISH software is designed to meet the requirements of a range historians pursuing a range of historical enquiry. It is flexible, conforms to those operating system standards generally pursued within the profession, and is easily generalisable across the most widely used machine types.

DISHData is a powerful and flexible data-entry program, designed to meet the needs of the historian who cannot always, no matter how thorough his preparation, predict the form original sources might take. It allows easy preparation of a data structure that can be changed to reflect unforeseen changes in data. Functions are incorporated to reduce the excessive amount of keystrokes required to capture repetitive data, allowing the scholar to reflect the full textual content of his source. Data can be written out from DISHData in any data format, allowing it to be used with virtually all available dbms - equally the program can be used as a conversion utility to convert data from one dbms format to another.

QUANAL is a program developed at DISH to allow simple quantification of data that is not primarily numeric. It is very similar to a report generator on a database package. QUANAL was originally written as a numeric enhancement to Quest, an information handling program developed by the Advisory Unit much in use in schools - hence its name (4). QUANAL sorts data according to some attribute (for example Occupation) and will produce statistics relating to numeric value associated with the attribute (for example, mean salary for each occupation). Typically occupations will be a straight transcript of those in the original source. Naturally the occupational descriptions will be inconsistent - one occupation will be described in a number of slightly different ways - or they will be imprecise - what is an Engineer, for example? QUANAL presents the user with a list of the occupations (or whatever attribute has been chosen) and their related statistics. The user can also select out particular items which can then be presented graphically, or group together items that seem to share similar characteristics and aggregate their associated numeric values. The emphasis is on the user making the decision on the basis of the original material presented to him on the screen.

QUANAL treats the data in an absolutely consistent way regardless of the source you are using. The same technique can be used for any database and as a consequence the learning curve for QUANAL is a very short one. However, the onus is on the user to decide which statistics are relevant - the program merely produces lists of statistics to choose from. To be general purpose, the analysis has to come from the human and not the machine.

DISHLink is a matching, or record linkage program currently under development. Like all current DISH products it is being written for use with MS WINDOWS, and takes full advantage of the opportunities afforded by this versatile and user friendly environment. Matching between databases and within databases, and determining whether Joseph Chamberlain in one place is the same man as Chamberlain, J. elsewhere, is one of the most fundamental tasks that a historian has to undertake (5). With or without the computer the historian is always in pursuit of people, ideas,

institutions (or whatever) that share the same characteristics. There is, however, no task that is more difficult. DISHLink uses an algorithm to determine a closeness of match and then prompts the user in borderline cases to use his skill and judgement to decide what is a match and what is not. Again many of the decisions have to be made on the basis of instinct and previous experience, characteristics which are found more often in humans than in computers. By necessity selection on the part of the user is built into DISHLink as it is into QUANAL.

The final aspect of the range of DISH software is DISHInfo, a WINDOWS based interface to a dbms that incorporates a query system, browsing facilities, notepad etc., and that when complete will bring together the various elements of DISH software, and offer paths through the WINDOWS environment to other commercially produced packages for presentation graphics etc. The way in which historians extract information from databases will be reflected in DISHInfo. This includes high hit rates, an emphasis on both analysis and extraction, and a strong interest in information that is related to retrieved items.

Since the DISH Project began it has become apparent to those involved that even the best software is unable to overcome a poorly organised database. This lesson was particularly brought home by a course taught each year to introduce students to the principles of historical computing (Computing for Historians). In one of the exercises, each of the students was given an identical source (matriculation records for nineteenth century students at the University of Glasgow) and told to produce a database. The result was an astonishing difference in effectiveness between the databases produced - and virtually as many data structures as there were students in the class. In some (perhaps many!) cases the database could be totally useless for any form of analysis whatever. Generally students with little experience of looking in detail at raw information had difficulty in identifying firstly discrete data items, and secondly identifying those data items that were relevant to the questions they had been prompted to answer.

This has obviously given the members of the DISH team that have been most closely involved with this teaching a useful insight into database construction. However, it is a frustrating way to obtain these techniques as the historian is required first of all to do it wrong, and then attempt analysis on his substandard database even to understand why it is so important to get it right. The historian must go through this process before taking on board various tips for improving his database.

Needless to say, this phenomenon is common to many learning situations and is not unique to database construction. Unfortunately, what is often not realised is that a substantial amount of data is required really to test a database. »Substantial« here means more information than can comfortably be manipulated if it were hold on paper (a suggestion would be a

minimum of two hundred records). Database construction is boring and very timeconsuming so the encouragement is there to get it right the first time.

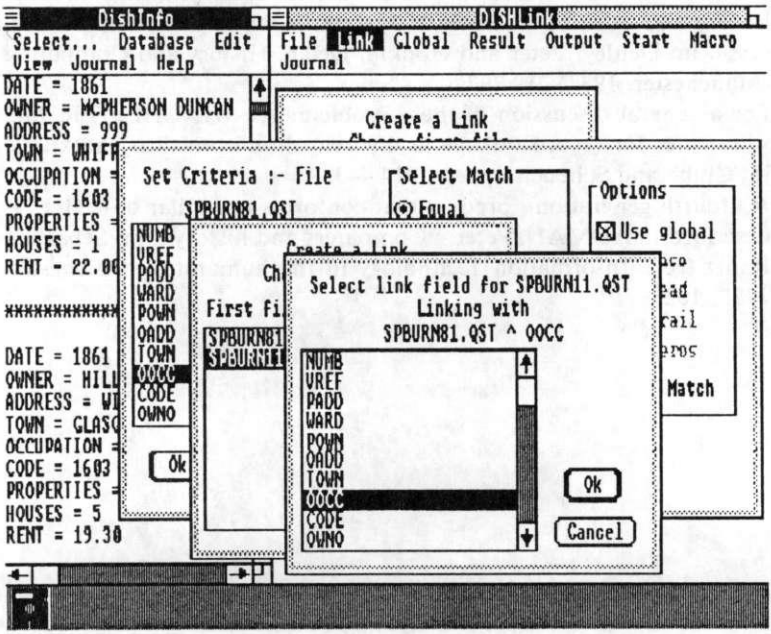
CBIS, a project that has been set up by DISH in conjunction with the British Government's Training Agency and the Clydesdale Bank, is now working on a piece of courseware to instruct on the issues of database creation. The product, again being developed in WINDOWS, will be based on a collection of business records relating to a prominent firm of Glasgow undertakers, and public records. The learner will not have to create his own database but will structure an existing relational database (the presence an structure of which will be hidden from the user) in a number of different ways. The potential offered by the graphics of WINDOWS will be deployed to display the data in a way which suggests no more than the physical structure of the data in its original volume. An expert-system will ask the learner at the start of a session what he hopes to achieve from the exercise - these objectives will be defined in terms of questions relating to the data contained in the courseware. The consequence of his decisions at the analysis stage will be sent back to the user by the courseware, which will indicate whether or not these objectives have been met. The hope is to allow the learner to make his mistakes before he has to start database construction in earnest. The courseware will also demonstrate that as far as possible organisational issues should not conflict with the need for maintaining the integrity of the source.

To sum up, to allow a widespread and effective analysis of historical databases the following conditions must be met:

- The data on the computer must reflect the integrity of the original source. Good organisation will allow a wider range of analysis to be carried out.
- Software should allow the data to be structured and searched in a way that will identify both the general and the particular. The data should drive the dbms, and not the other way round!
- The software must present a limited number of flexible tools to the user. These must be useful for as many datasets and forms of analysis as is possible.
- Software should be powerful, friendly (often said but rarely achieved) and intuitive - hence our decision to concentrate on MS WINDOWS, which offers a path to the greater potential of OS/2.

There are few software packages that offer the historians a complete solution to his problems. Software development at Glasgow has concentrated on creating a variety of tools that can be used singly or severally in a flexible environment (see fig. 1), with other proprietary products (6). The

Fig. 1: DISH Software: a variety of tools in a flexible and friendly environment...



objectives of the project are to produce tools designed to maximise the use the historian can make of machines without compromising his basic scholarly objectives and materials.

Notes

1. For the social scientist's preference for statistical packages over database management systems see STONE, Philipp J. »A perspective on social science data management« in: Clubb, Jerome and Scheuch, Erwin K. Historical Social Research: the use of historical and process produced data (Stuttgart, 1980) 444-454.
2. For a recent discussion of this issue see papers by BLUMIN, S.; HIGGS, E. and SCHURER, K., in: Richmond, L.; Mawdsley, E.; Morgan, N.J. and Trainor, R.H. (eds.). History and Computing 3 (Manchester, 1989).
3. For a general description of the DISH project see MORGAN, Ni-

cholas J. et al., »The design implementation and assessment of software for use in the teaching of history«, Historical Social Research, 38 (1986).

4. This package is described in WILD, Martyn. information handling, history and learning: the role of the computer in the historical process^ in: Denley, Peter and Hopkin, Deian. History and Computing (Manchester, 1987) 289-297.
5. For a general discussion of these problems see WINCHESTER, Ian. »Priorities for record linkage: a theoretical and practical checklists in: Clubb and Scheuch op. cit., 414-443.
6. A »fourth generation« product that conforms to similar objectives is described in ADMAN, Peter. »Computers and history«, in: Sebastian Rahtz (ed.). Information Technology in the humanities (Chichester, 1987) 102.