Automation on Parnassus: CLIO - A Databank Oriented System for Historians [1980]

Thaller, Manfred

Historical Social Research
Historische Sozialforschung

Manfred Thaller:

Automation on Parnassus. CLIO – A Databank Oriented System for Historians [1980]

doi: 10.12759/hsr.suppl.29.2017.113-137

Published in:

Historical Social Research Supplement 29 (2017)

Cite as:

Manfred Thaller. 2017. Automation on Parnassus. CLIO – A Databank Oriented System for Historians [1980]. *Historical Social Research* Supplement 29: 113-137. doi: 10.12759/hsr.suppl.29.2017.113-137.

Historical Social Research
Historische Sozialforschung

Other articles published in this Supplement:

For further information on our journal, including tables of contents, article abstracts, and our extensive online archive, please visit http://www.gesis.org/en/hsr.

Automation on Parnassus. CLIO – A Databank Oriented System for Historians [1980]

Manfred Thaller[*]

Abstract: *»Automatisierung auf dem Parnaß. CLIO – eine Datenbank-orientiertes System für Historiker«*. To support several ongoing research projects the development of a general purpose system was started in 1978 at the Max-Planck-Institut für Geschichte in Göttingen. This paper describes the second version of that system. Controlled by a simple scripting language, its most important features are: (a) a flexible input system combining free field and tag/content representation of data that can be structured in very complex hierarchies, (b) a retrieval system, (c) a system for the interactive coding of historical sources. (d) Various ways of entering codebook like "thesauri" provide a cheap way to recode as often as necessary material of doubtful semantics. (e) A system for nominative record linkage is provided which supports name comparisons by two classes of algorithms. Base algorithms can be adapted to suit the dialects of a particular area by a simple set of algorithm defining directives.

Keywords: Historical Data Bases, nominative record linkage, CLIO.

1. Introduction

Several research projects currently in progress at the Max-Planck-Institut für Geschichte in Göttingen use a number of different techniques of EDP.[1] The Computer

[*] Reprint of: Manfred Thaller. 1980. Automation on Parnassus. CLIO – A Databank Oriented System for Historians. In *Historical Social Research* 5 (3): 40-65. doi: 10.12759/hsr.5.1980.3.40-65.

[1] The following projects are currently using CLIO: David Sabeans research on the village of Neckarshausen, cf. David Warren Sabean: Verwandtschaft und Familie in einem württembergischen Dorf 1500-1870: einige methodische Überlegungen. In: Werner Conze (Ed.), Sozialgeschichte der Familie in der Neuzeit Europas (= Industrielle Welt XXI), Stuttgart 1976, 231-46. Three closely related studies on protoindustrialisation in several small communities by Hans Medick (Laichingen auf der Schwäbischen Alb), Peter Kriedte (Krefeld) and Jürgen Schlumbohm (a number of villages in the area of Osnabrück), cf. Peter Kriedte, Hans Medick and Jürgen Schlumbohm: Industrialisierung vor der Industrialisierung (= Veröffentlichungen des Max-Planck-Instituts für Geschichte LIII), Göttingen 1977. A Project by Albert Cremer on Die französische Magistratur von 1560-1610. A Project of Alf Lüdtke on Erfahrungen und Lebensweise von Fabrikarbeitern der ersten und zweiten Generation (Ruhrgebiet and märkisches Sauerland, 1830-1914) will probably reach the stage of data input in January next year. Besides these projects within the institute CLIO is at present used by the Kommission für Literaturwissenschaftliche Motiv-Themenforschung der Akademie der Wissenschaf-

DOI: 10.12759/hsr.suppl.29.2017.113-137

is employed to compute various statistical measures now familiar to many historians in such subjects as historical demography or social mobility. In addition, EDP is also used in a number of ways as yet not so commonly known: most of the projects at the institute deal with communities which are studied very intensively over a long period of time. To do so, practically all existing sources are linked together to form a kind of vastly expanded family reconstitution, reconstructing not only the biological families – as usually done in comparable studies which are primarily oriented towards the study of problems of historical demography[2] – but also socio-economic families. This implies that not just the registers of baptisms, marriages and burials are linked by means of the names of the people concerned, but that this primary reconstitution is connected further with tax registers, *Inventuren und Teilungen*[3], *Leggebüchern*[4], the records left behind by local courts[5], census registers an so on.

Dealing with such large amounts of source material the use of the computer is clearly advisable – though not without problems. Projects that operate on such a scale cannot be expected to reach their ultimate goals within a few months; therefore it is more often than not impossible to decide exactly at the time of data input which parts of a given source will be needed most at the time of analysis and which ones can be discarded. Even more complications are caused by the very different logical structures of the sources used: input formats, which could be used satisfactorily for a tax register, will in most cases be anything but satisfactory for the management of the minutes of a church consistory or another body of local government. The intention to use rather different techniques for the analysis of a given source introduces further complications: the minutes of a church consistory can provide insight into illegitimacy in more ways than one – e.g. by providing incidence rates which can be analysed statistically *and* by offering qualitative material about changing attitudes over time, which can be studied with the help of computerbased content analysis.[6] These different ways of dealing with the same materials – syn-

ten zu Göttingen. Data from several other research projects outside of Gottingen are presently made compatible. So the author himself will use a large set of data describing medieval pictorial sources made available by the Institut für mittelalterliche Realienkunde in Krems an der Donau (Austria) for a study on the applicability of statistical measures upon such material.

[2] For the analysis of such reconstitutions see Arthur E. Imhof and Thomas Kühn: Die Analyse kirchlich-administrativer Daten mit Hilfe der EDV. In: Heinrich Best and Reinhard Mann (Eds.), Quantitative Methoden in der historisch-sozialwissenschaftlichen Forschung (= HSF III), Stuttgart 1977, 11-64. Available at <http://www.gesis.org/hsr/archiv/buchreihe-historisch-sozialwiss-forschungen-hsf/hsf-3>.

[3] Marriage contracts and post mortem inventories. For this type of source have a look at Heilwig Schomerus, Die Arbeiter der Maschinenfabrik Esslingen, Stuttgart, 1977, 279 and passim.

[4] Register of all sales of linen in the Osnabrück area.

[5] On this material see David Sabean as quoted in footnote 1.

[6] For a survey see Ekkehard Mochmann, Computer Aided Content Analysis of Historical and Process Produced Data: Methodological and Technical Aspects, in: Jerome M. Clubb and Erwin K. Scheuch (Eds.): Historical Social Research (= HSF VI), Stuttgart, 1980, 414-30. Available at <http://www.ssoar.info/ssoar/handle/document/32617>.

chronously if feasible – are usually supported by computer programs needing as input data of completely different formats and requiring the knowledge of control languages that have scarcely anything in common.

To solve these problems the author started 18 months ago the development of a program system – named CLIO – with two main perspectives:

1) Historical sources should be accepted in an extremely flexible format as close to the original as possible and providing a maximum of economy. Clearly these aims are conflicting; using the system you have still to decide which one is of greater importance to you, but both of them are supported, so that you can use CLIO to manage a source that has been transcribed literally – and is possibly printed later directly from this form of input – as well as to translate numerical values from a register containing a large number of different systems of measurement usually not decimal quickly into something more compatible with the requirements of a computer.
2) This material, then, is worked upon by a system that can administrate it – performing linkages of files via names of persons mentioned therein, merging data structures of almost unlimited complexity and so on – and perform a number of analyses itself, and can “bring you in touch“ with other programs – either by direct interfaces or, at least, by preparing your data to suit optimally the input requirements of another program. The first advantage is that for all of this you use one control language instead of having to know the detailed formats of 20 different types of parameter cards.

Not all these objectives have been reached within 18 months, of course; still, the version of the system that is currently being tested goes quite far on the path just outlined. This version of the system is the subject of the paper presented here, though a few references to further objectives have been entered where they are going to be implemented in the next stage.

Before this detailed description follows, a few words should be said though, on the general ideas behind the concept of computer use by historians presented here. The first – seemingly trivial – one is that “using EDP“ tells something about the research tools and the techniques of work someone employs; it is not, as such, considered a constituent of methodology. One of the techniques that is supported by – and indeed often not possible without – the descendants of ENIAC is the one subsumed under “quantification“, the use of statistics. One might reasonably go further: computers are admirably suited to that type of work. Still, in the opinion of the author, turning the contents of a historical document into numerical codes and using SPSS or some other program to compute a series of statistics from that material, are not necessarily any methodological *nouveauté* – deterministic and anecdotic thinking can easily be expressed in statistical terms. Therefore CLIO, as described below, goes quite far to remain neutral; the system can be used to administrate a card file of gigantic dimensions on the one hand and on the other provide access to statistical programs. This later function is available now and in the near future the system will support such sophisticated methods as loglinear modelling on its own.[7] In the background of CLIO there are a large number of methodological considerations – indeed the data structures of the system could be described as a series of set theoretical equations designed to formalize a theoretical concept of what “information“ in historical sources is. As presented here, though, it is simply a tool for historians – and should therefore be discussed only at this level. Still, even forging a tool, you have to have some idea how it is going to be used. Generally speaking, CLIO will be suited best to strategies that have a logic similar to this one:

1) The design of CLIO assumes that historians – different from (other) social scientists – formulate their detailed questions in a much more intensive preliminary scanning of the material that shall ultimately answer them. So the system provides a large number of tools for different methods of data retrieval, without asking for much coding of the material to be used.
2) It is further assumed that one of the aims of historical research is, to generalize beyond the description of single cases. Tools to define and analyze similarities are provided and this aspect will be subject to considerable development in the future.
3) Finally it is taken for granted that one of the most substantial tools of historical research is the comparison of structures in sources from different times and places. In the long run it is hoped to support this by presenting a system that allows the simultaneous analysis of material from very different sources, strongly supporting secondary use of anything once made machine readable.

2. Preparing the Data

Leaving such lofty views we very soon reach the drudgery of the day-to-day use of the computer. The most time consuming part of it is usually the stage where a historical source is turned into a machine readable file.[8] How you do this physically is

[7] This is made possible by an improved version of Goodman's ECTA made available for use within CLIO by the VASMA project in Mannheim of the Volkswagenstiftung.

[8] Input systems as the following one have been discussed very much during the last years. Without counting them systematically, the author thinks that in the last few years in the Historical Methods Newsletter (now under the title Historical Methods) alone about 15 papers have been presented that describe input systems. (The situation is complicated even more, as almost any publication of results gained with the help of the computer contains a short description of the forms of input used. While mainly explanations what a FORTRAN format is for the benefit of colleagues who know less about computers, many such introductions contain valuable ideas.) Even the attempt to give an adequate bibliography of this publications would have to include about 100 bibliographical items. So, though CLIO is very much indebted to ideas people had before, only three systems can be mentioned explicitly. Roger Schofield and Ros Davies: Towards a Flexible Input and Record Management System. In: Historical Methods Newsletter 7 (1973/74), 114-24. C. J. Jardine and A. D. J. MacFarlane, Computer Input of Historical Records for Multi-source Record Linkage, in: Michael Flinn (Ed.), Proceedings of the Seventh International Economic History Congress, Edinburgh, 1978, Vol. II, 71-8. Marcel Couturier presented a paper on “A Method of Data Collection and Processing: The FORCOD System” at the 1977 conference of QUANTUM and SSHA which has

largely dependent on the resources which are available at a given computing center and for a given project. Most data are still punched on cards – though this is scarcely optimal, particular for large projects. In the literature on computer use by historians numerous more efficient ways have been described to transfer information onto machine readable media: writing things onto OCR compatible sheets[9], talking into a dictaphone and letting the tapes be transcribed by a data typist[10], taking the original source and transferring its contents directly into mass storage by means of a video terminal[11] or even taking a small terminal with a micro processor and a storage medium into a data archive.[12] All of these methods are compatible with CLIO, which is only concerned with the logical, not the physical structure of the data. As it might make clumsy reading to write everything down while considering half a dozen different media, it is assumed in this paper that sources are transcribed onto forms by hand and afterwards entered either via video terminal or punched cards.

Mentioning the very necessity of transcription probably already clarifies why in most applications the formula "one source becomes one file" will hold.[13] Indeed most of the time one particular bundle of relatively similar sources – a marriage register, the' minutes of a court, a series of medieval documents, or a tax register will be turned into a distinct CLIO system file. The largest unit of such a file (approximating the "cases" of SPSS), we will call a "document" in this paper, representing more or less one distinct unit in our source: a marriage entry, a family in

not been included in the publication of those papers in HSF volume 6. A good impression of the practical use of this system can be gotten from the article quoted in footnote 10 below.

[9] E. g. Ingrid Batori, Sozioökonomische Untersuchungen in Franz Irsigler (Ed.), Quantitative Methoden in der Wirtschafts- und Sozialgeschichte der Vorneuzeit (= HSF IV), Stuttgart, 1978, 24-42. Available at <http://www.ssoar.info/ssoar/handle/document/32555>.

[10] E. g. J. Dupaquier, Quelque Reflexions sur l'utilisation de l'ordinateur par la reconstitution automatique des families. In : Bulletin d'information sociétê de Demographie Historique (No. 22, Octobre 1977) 12.

[11] This method is currently being tested at the institute using a micro computer. In the case of marriage registers the efficiency is about 80 percent above the old source-to-paper, paper-to-video-terminal method.

[12] This method we will have to use in the near future, as several sources needed cannot be taken out of the respective archives. While the interest to use the equipment after the end of this phase together with the micro computer we already have will probably lead to the selection of a slightly more expensive configuration, it may be of interest for other researchers that the cheapest offer for the combination of videoterminal and storage medium was priced at about DM 3500. Hardware becoming cheaper, this way of input should not be overlooked in the future.

[13] Edward Shorter introduces his chapter on "Processing the Data" in "The Historian and the Computer" with an argument against superficial use of technical language. Particularly he speaks against the use of the term "data base" and asks historians to "call the information abstracted from historical sources the 'data file'" While sensible in what already might be called the heroic age of computer use in history this terminology is simply misleading in the context of larger projects of today as described here. When a "file" is mentioned in this paper this term designates the same thing as usual in data processing, i. e. a entity of data stored separately on some medium. The "information abstracted from historical sources" usually makes up dozens of such files and is summarized as the "data base of a given project".

census returns, the minutes of one case of a court, one marriage contract or post mortem inventory, of a collection. The physical size of a document may vary considerably: currently used system files include taxregisters, where one document represents one line of the source and a series of *Inventuren und Teilungen* where what becomes one “document“ may fill 40 or 50 pages in the original source.

Such a document, particularly the larger ones, looks first as a vast unordered heap of information, related to possibly 30 people, 4 municipalities and 40 years. To make this manageable, we order this information into “groups“, every group describing a subunit of the document that can appear as often as necessary. A census return, for example, could be described as consisting of a group named “head of household“, another one defined as “wife“ and several others – “children“, “servants“ and so on. A document of this sort might consist of just one of these groups or of 30, describing the household of a rich peasant. (A restriction actually exists: should anybody try to analyze the family of an Arab Emir with more than 1023 children and/or more than 1023 servants, CLIO might be troubled, though there are ways to bypass such limitations. Usually such limits as exist are about as relevant for the day-to-day user as the one that makes it illegal to assign a text of more than about 80.000 characters as value to a given variable). Within these groups it will make our analyses easier to differentiate between several “elementary items of information“ – surname, occupation, origin and so on; that is, “elementary items“ are within the context of CLIO approximately the same thing, as “variables“ in the context of SPSS. Using these concepts we might transcribe a particular census return as follows:

```
HEAD$SURNAME=SMITH/1ST.NAME=JOHN/OCCUPATION=CARPENTER/
ORIGIN=SOMEWHERE
WIFE$MAIDENNAME=MILLER/1ST.NAME=MARY/ORIGIN=SOMEWHERE
ELSE
CHILD$1ST.NAME=FRANK/SEX=MALE/BIRTHDATE=12 MAR 1746
CHILD$1St.NAME=JUDY/SEX=FEMALE/BIRTHDATE=19 SEP 1749
/REMARKS=ABSENT
```

Clearly this is not a very efficient way of entering data. We might improve performance by simply telling the machine to consider the fields as defined by the order of their appearance, writing for example:

```
HEAD$SMITH/JOHN/CARPENTER/SOMEWHERE
WIFE$MILLER/MARY/SOMEWHERE ELSE
CHILD$FRANK/MALE/12 MAR 1746
CHILD$JUDY/FEMALE/19 SEP 1749/REMARKS=ABSENT
```

Unnecessary to mention that we might reduce further our transcription work, if we would replace our SOMEWHERE’s by abbreviations – how far this is possible within a given source, can only be decided by the researcher. More easy to devise are such abbreviations as in the case of sex, where we have only two possibilities. Reducing further the length of our data names – in this case using CLIO’s capability to take almost anything, including the Null String, as data name – we might come to something like

```
H$SMITH/JOHN/CARPENTER/SOMEWHERE
W$MILLER/MARY/SOMEWHERE ELSE
```

```
$FRANK/M/12 MAR 1746
$JUDY/F/19 SEP 1749/REMARKS=ABSENT
```

This example was drawn out so long to make clear that we do not describe the way in which we would necessarily input certain data but rather the class of formats that can be handled by CLIO. For the system the three ways to transcribe data are more or less equal. In day-to-day work we could find reasons for using any of them. (Even for the clumsy first one. It loses a lot of its clumsiness, when we remember that it allows full use of dictaphones as fervently advocated in the context of FORCOD.[14]) When during further examples we use a drawn out notation, we do so simply because it is largely self-documenting.

When we speak of a system that has a user-friendly control language, we quite clearly need something that is an equivalent to the "formats" of more conventional systems. This is provided by the STRUCTURA command of CLIO's command language. In our example it might read

```
STRUCTURA
HEAD(SURNAME:LOCUS,1St.NAME:LOCUS,OCCUPATION:LOCUS,
ORIGIN: LOCUS,
WIFE (MAIDENNAME:LOCUS, 1St.NAME:LOCUS,ORIGIN:LOCUS)
CHILD(1St.NAME:LOCUS,SEX:LOCUS:CONDICIO=FM,BIRTHDATE:
TEMPORA))
```

This would be the form, we would need for the second way of transcribing the date, at the same time abbreviating sex. Every :LOCUS means that the system shall use the name out of the STRUCTURA command in all cases, where the construct "name=" – which we will call a tag from now on – is not contained in the data. The :TEMPORA indicates that this field contains a calendar date, the :CONDICIO=FM announces a field that contains only single letters, with nothing legal but M and F. A further function of this command is to describe the relations between different groups within a document. In our case this means that every CHILD is considered as related to the HEAD (of Household) so that we can always draw its surname from the information contained in the group HEAD, and that every CHILD which appears in the data is logically equal to all other children of the same family. Which children belong to which Head of Household is simply determined by their physical appearance – that is, every Child is considered to belong to the last Head of Household encountered. This structuring is not very spectacular in our example, but it can be very important in practical work, as such hierarchical files[15] of course minimize the repetition of single items of information. In one of the files currently being handled by CLIO, this STRUCTURA command runs across more than 200 lines including groups as "the father of the partner of the illegitimate child of a child out of a given marriage of the head of the household". (Taking the group indicating the start of a new document as level 0, structuring may at present use as many as 9 levels.)

[14] See footnote 10 above.

[15] For the reader slightly more familiar with data base management it may be mentioned that CLIO internally uses a combination of hierarchical and relational principles of data organisation.

Additionally we have the possibility to define a whole series of plausibility checks within such a STRUCTURA command. If we add a construct of the form :SOLUM=3:SEMPER=3 to the name of a given group within the structure, we will be warned every time the program finds a case where this group appears more or less often than three times in the one in which it is logically included. Furthermore we can check if calendar dates are in sequence and several other things.

Particularly this possibility to check the plausibility of the data – marriages with no brides concerned and so on – makes the STRUCTURA command very important. We need not bother, though, to include every elementary information that may appear in a given source through 200 years in this command, as CLIO uses a dynamic format; that is, if a tag is encountered in the data being input, it is checked to find if it is already contained in the internal representation of the STRUCTURA we formulated; if not, it is added, and a list of all appearances is automatically printed at the end of the job, to provide, for example, a check against accidentally misspelled tags. "The job" just mentioned is the one that creates a CLIO system file out of input data, as, unsurprisingly, the system does not attempt to handle the data in the form just described, which is quite well suited to most sources, but scarcely an optimal representation of the data from the machine's point of view.

Such a converting job need not convert all the data you want to analyze at once – it is perfectly possible to add data to an existing system file. It is even possible to change the formats – as described in STRUCTURA commands – while you transcribe your material; a form of input convenient for a baptism register in 1690 usually is anything but convenient for the register of the same location in 1870. If you encounter such a problem, you simply divide your data in as many different parts as necessary to transcribe everything with the least effort and call subsequently one of the CLIO merging routines. If you want to merge two registers

```
CONNECTIO PRIMUM=REGISTER1,
QUE SECUNDUM=REGISTER2,NOVUM=BIGREGISTER,
QUE MODUS= INCREMENTUM
```

will do the job, calling a rather complicated set of programs, which first computes the logical sum of the STRUCTURA commands of your two system files and modifies all internal addresses so that the two sets of data can finally be physically merged. Disambiguation of ambivalent tags (say you used the possibility to use 1-letter abbreviations for a given elementary information in REGISTER1, but not in REGISTER2) is among the services performed automatically.

Even more than by these possibilities on the level of file management, the actual work of transcribing sources is smoothed by the definition of an elementary information. Basically CLIO does not assume that historians deal with clear and sharp-cut information. Thus multiple entries are legal without any conscious effort – if you write OCCUPATION=CARPENTER;FARMER you have later on the possibility to use CARPENTER and FARMER as distinct values of this "variable" – e.g. you can translate this elementary information into a set of variables JOB1 TO JOB10 for analysis in SPSS – or treat "CARPENTER;FARMER" as one value.

Additionally you have the possibility to use "subfields" to every elementary information. An entry like

```
OCCUPATIQN=BLACKSMITH#WHAT DID THE POOR BRUTE DO IN SUCH
A SMALL VILLAGE ?, %BLAIKSMITYE
```

would be perfectly legal. Whatever you write after a number sign (#) becomes known to CLIO as a “comment“ to the elementary information where it is found, while everything after a percent sign (%) is considered as “original wording“ – these names of course being only mnemonics, indicating how one might wish to use subfields.

Whatever is written in front of the first of these two symbols – or the whole elementary information if none of them appears – is considered as the “basic information”. It can have any one of the following 4 datatypes.

2.1 Lingua

This datatype describes the meat-and-potatoes of the system. It simply announces that you are entering a text of arbitrary length, consisting of any symbols not specifically reserved by the system. As it is completely your decision what you put into basic info’s of this type, it is unfortunately impossible to find out that you entered something illegal- semantic checks are rather far off.

2.2 Tempora

This datatype allows the efficient treatment of calendar dates. You can enter them in any of the following forms:

14.7.1263 (considered to be the “German representation”)
14 JUL 1263 (considered to be the “American representation”)
PRI ID JUL 1263 (called “Latin representation”)
14 FLO 7 (14th Floreal in the seventh year of the republic; the “Revolutionary representation”)

Additionally you can enter an interval instead of a single date, where you simply give the *termini post* and *ante quem*, if, as is often the case, the exact date is unknown.

```
BIRTHDATE=17 MAR 1657-23 MAY 1658
```

would be perfectly intelligible.

Of course data of this type are checked, before they are entered into the system file – you will be warned of 29th of February 1800 and similar dates. The introduction of the Gregorian calendar will be considered, though, unfortunately, the system has no way of knowing it, if you don’t specify it when checking and converting your data with a suitable command.

2.3 Condicio

This data type considers every letter or digit in the basic information as a distinct multiple entry. This is an extremely economic way of entering information, where only a few values are possible, say sex or religion. This even the more so, as you can put these two things quite easily into one elementary information, using FL or LF for female Lutherans, CM or MC for Calvinist males and so on. CLIO provides a set of ways how such composite items can be easily spread over more than one variable, when you want to use such data with statistical programs, where multi-purpose-variables, while convenient for input, create a lot of trouble.

2.4 Numerus

This is, strictly speaking, not yet a data type on its own within CLIO, but will become so rather soon. Presently the following ways of entering numeric information already exist, but are at the beginning treated as LINGUA data. That does not make much difference for the commands a user has to give, but it is responsible that CLIO at present is not efficient for use with data which are strictly numeric. The following rules allow, though, a very flexible input of numeric information that is contained in a source in systems of measurement not being decimal. Prices in pounds, shillings and pence could for example be entered *either* as a "Triplet" – say 2.11.3 – *or* as a "series of qualified numbers" – e.g. 2-POUNDS 11-SHILLINGS 3-PENCE. While this second notation seems incomparably clumsier, it can be very useful, when you intermingle both notations. If you have a source, where 60 % of the prices are given in one currency system, 35 % in another one and the rest is priced in assorted coinage – Louisdor and other coins that were used throughout all of Europe – you can define the following "thesaurus" to handle this problem.

```
THESAURUS PRICES
TRIPLEX= 240/9999.12/19.1/11
SECOND.CURRENCY=117
LOUISDOR=123
EXITUS PRICES
```

If, after defining such a thesaurus, you translate 2.13.2 for a statistical analysis, you will get the value in pence, if you translate 2.2.13 you will get zero and a warning message (the figures after the slashes in the TRIPLEX directive being the legal limit); 2-SECOND.CURRENCY will result in 234 and 5-LOUISDOR will be treated as 615 pence (the misjudgement in the guess I made of the relation between pound and Louisdor is solely my responsibility, not the computers).

Besides this you can – in a limited way – enter directly complex expressions out of sources; provided you define the correct thesaurus, the remark that a given heir gets 1/3 of the sale of 5 *Maltersaat*, 2 *Scheffelsaat* and 13 *Quadratruten Land* at 1 *Taler*, 4 *Silbergroschen* and 3 *Pfennig* for each *Quadratrute* could be transcribed as

```
MONEY= (5-M 2-S 13-Q)*(1.4.3):(3)
```

- admittedly a not very simple expression, but still one that is written down faster than it takes to compute the numeric value.

3. Retrieving the Data

Having input the data, one usually wants to get them back from the machine; indeed, about the minimal requirement a program system can be asked to fulfill. The most simple way to do so is to ask CLIO to give a description of the system file, or, formulated in the CLIO command language, the single word DESCRIPTIO. This can result in thousands of pages of output as the very compressed raw input will be formatted to insure maximum readability. Indeed this command will usually be given only once for any particular dataset, providing the researcher with an exact description of the material he is going to use. This not very spectacular command can be influenced to give descriptions of subsets of the data only – e.g. you can get a detailed printout of all those of your documents, which are situated within any period of time. This modification is asked for by the following command:

```
QUAERO DATE(MYFILE)=1.1.1701-31.12.1710
```

meaning in plain English: “I’m asking for all those documents, where the basic information labelled ‘date’ (which has to have datatype TEMPORA) takes any value between the 1st of January 1700 and the last of December 1710”. It is of course much more flexible than this

```
QUAERO SURNAME(FATHER(BRIDE(MARREGISTER) =
QUE SOUNDEX(SMITH/MYSOUNDEX/PRIVATUM=MYLIBRARY)
QUE VEL PRINCIPIUM JONES LIMES
```

would have to be translated into English as follows: “I’m asking for all those documents in the file MARriage REGISTER where the surname of the father of the bride has either any spelling that can be reduced to ‘Smith’ if one follows the rules of an algorithm of the SOUNDEX class which I laid down under the name MYSOUNDEX in my private library of algorithms named MYLIBRARY, or is given exactly as ‘Jones’.” (For the meaning of SOUNDEX algorithms see below under the heading “Linking the Data”).

Let’s look more systematically at this command! The first thing we find is the construct SURNAME(FATHER(BRIDE(MARREGISTER). Expressions like this are called a “position“ within the context of CLIO. You can imagine them as the equivalents of a variable name. A bit more is accomplished, though, by writing them down as would be by giving the name of a variable within other comparable systems. The term MARREGISTER in the innermost bracket is used by CLIO to access a file under this name. That means, in most cases you can forget about the control language of the operating system, as the link between the external file and the logical file name is constructed without your explicit interfering.

The part of the construct to the right of the equals sign is called a comparison pattern that is, the position you have named has to be successfully compared to that pattern to cause the actions, which are specified by your further commands, to be brought about for this particular document and/or information group. The rules for their construction are somewhat beyond the scope of this paper; it may suffice to say that they include the logical AND, OR and NOT which are commonly used in such contexts. More than one of these constructs can be used with one QUAERO command of course; the command

```
QUAERO OCCUPATION(PERSON(DATASET)=NULLUS,SEX(PERSON() =M
```

could be used to get – via the keyword NULLUS, indicating something like MISSING in the statistical sense – information about male persons, where no occupation was given in the source.

QUAERO of course may not only precede DESCRIPTIO, but any of the more complex retrieval commands that follow. The most simple one, next to DESCRIPTIO, is the SCRIBE command.

```
QUAERO ITEM(DATASET)=SOMETHING
SCRIBE ITEM(DATASET)
```

being the CLIO way of expressing the classical task for any retrieval system: “If a certain field within my file contains a certain value, write it out.“ Formulating this and other retrieval commands, you have not to know very much about the logical structure of your file, i.e., you have not to bother about the relative positions of the field that shall be tested and the one that shall conditionally be written out, within the hierarchical structure of your data. Unlike quite many retrieval programs CLIO takes the view that every question, if not explicitly restricted to a narrower scope, takes its broadest meaning possible.

```
QUAERO OCCUPATION(CHILD(MARRIAGE(FAMILIES)=SOMETHING
```

will be fulfilled by any of the unknown number of children of the unknown number of marriages of the unknown number of families within your data file. This may, under some circumstances, lead to results that were not being asked for, but it reduces in most cases quite considerably the effort in thinking up and writing down your commands.

More complex than the simple “write out“ (SCRIBE) is the command that produces the registers you may want to use with a large set of data.

```
INDEX OCCUPATION(CHILD(HEAD(FAMILIES),
QUE OCCUPATION(RELATIVE(HEAD(FAIILIES),
QUE OCCUPATION(HEAD(FAMILIES)
```

or – using a bit of CLIO shorthand –

```
INDEX OCCUPATION (CHILD/RELATIVE(HEAD(FAMILIES),
QUE OCCUPATION(HEAD ()
```

would result in an alphabetic register of the occupations that appear in your data.

```
INDEX OCCUPATION (CHILD(HEAD(FAMILIES) CUM
QUE OCCUPATION(HEAD()
INDEX OCCUPATION (CHILDCHILD(CHILD(HEAD() CUM
QUE OCCUPATION(CHILD(HEAD()
CONTINUATIO
INDEX OCCUPATION. (FATHER(WIFE(HEADO CUM
QUE OCCUPATION(FATHER(HEAD ()
INDEX OCCUPATION(FATHER(SPOUSE(CHILD(HEAD() CUM
QUE OCCUPATION(HEAD()
```

would get you two independent two-column registers, the first containing an alphabetical index of all combinations of jobs in son and father generations, depicting thereby occupational mobility, the second in a similar way showing marriage mobility.

```
INDEX HOUSE-NR(TAXLIST):DEXTRA CUM SURNAME() CUM
QUE 1ST.NAME() CUM QUE AMOUNT.PAID()
```

might be used to recorder a tax list for comparison with some other source. The DEXTRA will insure that the numbers of the houses are entered right justified.

A problem may arise out of the treatment of multiple entries: if, in datatype LINGUA, a semicolon is encountered, the content of the elementary information is split and two distinct entries into the register are prepared. If you use 5 CUM's resulting in a 6-column register a combination of 6 basic items with one semicolon each would lead to 32 lines in your register instead of one, as every possible combination would be entered. To solve such problems you can add the option PLURAQUE to your index command; this might look like

```
INDEX PLACES-MENTIONED(DESCRIPTION(STORIES) CUM
QUE PEOPLE-MENTIONED(DESCRIPTION():PLURAQUE
```

If you place the command NUMERUS instead of INDEX in front of any of the above examples, you would get a count of all terms/combinations of terms that exist in your dataset, as specified by the positions included.

A more refined method of retrieving information is provided by the DISTRIBUTIO command, plotting the distribution of events in temporal intervals. A typical application would be:

```
DISTRIBUTIO BIRTHDATE(PERSON(DATASET):CYCLUS=MENSES,
QUE ANNI=50,TABULA=ABSOLUTA
```

This command would ask for a series of plots showing how the BIRTHDATEs of the PERSON's in your DATASET are distributed among the 12 months of the year. A different plot would be printed for every 50 years. (The program would take care automatically that those 50 years are selected to form half centuries, irrespective of the date of the earliest birth encountered; so you need not to be afraid to have to compare the distribution between 1717 and 1766 with that of an interval that's similarly "natural" for an historian who has been trained to think in centuries and fractions thereof.) The Plot will consist of bars that compare the frequencies, with which the different months appear (TABULA=RELATIVA would have provided you with plots that show the deviation from the mean occurrence instead).

One of the shortcomings of CLIO undoubtedly is that at present it does not provide any proper routines for content analytical approaches in the usual meaning of that term. Since many programs for such applications already exist the author hopes to get access to some of them, instead of reinventing the wheel. A not completely powerless feature is provided however. Look at the following commands which assume a CLIO system file consisting of the minutes of some local court, where SUMMARY(TEXT() contains an abstract of the proceedings.

```
QUAERO SEX(DEFENDANT(MINUTES)=M
LINGUA SUMMARY(TEXT( ) :NOMEN=MALES
CONTINUATIO
QUAERO SEX(DEFENDANT ()=F
LINGUA SUMMARY(TEXT():NOMEN=FEMALES
COMPARATIO PRIMUM= FEMALES, SECUNDUM=MALES
```

This 6 lines would result in:

- 2 files that contain the content of the various SUMMARIES broken down into words (which characters are to be used as separators is under control of the user). Such files are going to be the base of an interface into some more sophisticated system of content analysis, as soon as access to such a system that can be brought into operation at the machine we are using is gained.
- a printout that compares the frequency of the words in those two files, thereby comparing the language used in minutes dealing with male and female de-

fendants. (Options provide – among other things – the possibility to print a comparative concordance of all words along with this.)

A major improvement of the results is possible by use of the following construct:

```
EXCLUSIONES GERMANIAE,ANGLIAE
COURT
OBERAMT
EXITUS
```

If these lines would be set in front of the two LINGUA commands of our example, the results would contain only those words not in the list between EXCLUSIONES and EXITUS. Only two appearing that seems to be rather trivial: GERMANIAE and ANGLIAE on the first line ask though that this list of explicitly excluded words, is to be increased by all appearing in the predefined lists of supposedly not very interesting ones for the user – articles, auxiliary verbs and so on – that are, for German and English, administrated by the system under the names given. Writing INCLUSIONES instead of EXCLUSIONES would result in a file of only those words which are in the list.

4. Translating the Data for Statistical Analysis

Data as provided by CLIO contain potentially as much information as the original source did. For statistical analysis this wealth usually has to be reduced to the amount that can be handled by statistical programs as SPSS. This reduction is what is commonly known as coding. As every researcher knows, the definition of codes needs a more than superficial knowledge of the terminology used by an individual writing long ago. Indeed, if you have ever been working in the context of a research program, where one or more of your colleagues had some doubts about the usefulness of quantification in general and the sort of it based on computers in particular, you will know that the "yes, buts“ you are going to hear from them before codes acceptable to all of you are defined quite often lead to the result that it would have been easier to do everything you did if Signore Galvani had never seen a frog outside its pond. Still, it can be even worse, if you never heard any "yes, buts" at all – there is more than one project which produced hundreds and thousands of punched cards in its early stages while the results were finally computed by paper and pencil when somebody discovered that the most important aspects of the source were unfortunately forgotten by the person who designed the code and/or overlooked by the student coding the materials.

The first of these problems can be smoothed considerably if you have means to scan easily through your material to find out about the aspects your doubtful colleague raised. To enable you to do so, the retrieval routines of CLIO are very useful – indeed quite a few of their features were explicitly designed to be of assistance in that process. The second of these problems can disappear if you use the machine for your coding and it is more or less immaterial how often you repeat the process of turning your doubtfully worded source into the clear coding scheme you finally develop; though, by restricting the items of information you take into your CLIO system file at the beginning, you can destroy its usefulness as effectively as by coding it immediately in a high-handed manner. If you have once overcome this initial problem though, you have a lot of possibilities to translate the information contained in your system file into numerical codes.

The first thing you have to do (if you won't be satisfied by the defaults) using the present version to convert parts of a CLIO system file into a set of numerical codes is to define the frame of that translation process as a whole. Basically that means, you have to decide how the system shall solve the problem of translating the structures of CLIO into the flat "cases" preferred by statistical programs. How that is done efficiently is a rather complex subject. In most cases you will simply have to select one of two possible modes of translation.

TRANSLATIO MODUS=REPETITIO

tells the system that you want to use "cases" that are as similar to each other as possible. If the branching of your hierarchy does not allow the translation of a whole document into one case, as few varying types of records as possible will be created. Say you have data, which consist of families with one and only one head of household, who may have been married several times and may have an unknown number of children out of these marriages. In this case the system will provide you with cases of only one type, where

- every case contains all numerical codes that are the result of the translation of the information known about one particular child,
- every case contains all numerical codes that are the result of the translation of the information known about the respective mother,
- every case contains all numerical codes that are the result of the translation of the information known about the head of household,
- every case contains the necessary variables to differentiate the families, either using the REPORT facilities of SPSS releases 8 *sqq*. or means more simple.

If your families should contain additional information about the relatives of the head of household and there may be an unknown number of them, two types of cases will be output, one of them containing information about one particular relative, the head of household and the family as a whole, while the other is equally structured as the ones just mentioned. These two types will be differentiated by a system of control variables provided automatically by CLIO. All of them will have an equal number of variables, wife- and children related variables being assigned missing value codes for all cases containing values related to relatives and vice versa.

If instead of REPETITIO you would have asked for STRUCTURA, every group of information would be represented by one case plus the necessary control variables – a method clearly saving machine resources but requiring more careful thinking of the user who in every case will get an exact description of the contents of the resulting file that should enable him to use the result of a particular translation even if he does not quite understand the intricacies of its general logic. Of course for the more refined user various options and parameters are provided to control the details of the structuring of the resulting file. A rather simple one of these is the parameter DESTINATIO of the TRANSLATIO command given above that may close its discussion: if you specify DESTINATIO=SPSS you get a numerical file that can be read by SPSS by INPUT FORMAT BINARY. As the variables in the file description are numbered sequentially, the two commands.

```
VAIRABLE LIST VAR1 TO VAR96
INPUT FORMAT BINARY
```

were all that was needed in one case to create a system file in SPSS, containing quite a lot of information, each variable being described by its number in the 6 page protocol produced by the translating job. (Using DESTINATIO=INCERTA instead will provide you with punched cards in a neutral format, which can be read in easily by other programs.)

While this command and its power is quite decisive for the practical use of the CLIO translation routines, these routines themselves are probably more immediately interesting. Three ways of translating information are provided right now:

4.1 Translating by Codebook

This way of translating information will usually be selected for elementary information containing a large number of relative short terms, such as information about the occupation(s) or the origin of someone. In most cases one will start with the creation of a card index of the terms occurring.

```
CREATIO OCCUPATION(PERSON(DATASET)
```

would provide the user with a set of punched cards containing:

- empty columns to receive the numeric codes,
- optionally a running number that can be used as a separate code, which will enable the user to differentiate between different terms within his statistical file even when he has decided to collapse these differences in his code system,
- one term as given in the source data.

At the same time a register of all such terms is printed (usually at this stage a couple of overlooked punching errors will be discovered – such as TAXLOR instead of TAYLOR for example). Treatment of multiple entries is under user control.

CLIO, which in its design considers punched cards as a relatively obsolete medium, as long as video terminals and more sophisticated media of storage exist, falls back upon them in this case, as they are admirably suited to be reordered by hand, transferred into the library where you have to look up a couple of particularly obsolete terms, spread around your living room (consent of spouse supposed), while you try to get them into a systematic order and so on. Finally you should write the codes, you decide to use, upon them, feed them into a card punch and transfer the codes into the empty fields; these fields being at the beginning of the card, the process is quite quick.

This done, the actual translation looks like this:

```
INTERPRETATIO OCCUPATION(PERSON(DATASET):
QUE THESAURUS=MYJOBCODE
```

with the following lines somewhere in the vicinity:

```
THESAURUS MYJOBCODE
```

Here follow the punched cards described above; their number, by the way, is unlimited.

EXITUS MYJOBCODE

Translations of this type result usually in two variables, one of them containing the systematic code you developed, the other the running numbers produced by the machine during the execution of CREATIO.

4.2 Translations of Conditions

A translation of this type will usually be selected to translate either the presence/absence of a certain part of a word or the presence/absence of a word in a longer portion of text. In the context of German sources this will typically be used to translate the parts LEHRLING, GESELLE and MEISTER (apprentice, journeyman and master) of the composite nouns representing the occupational information. Here one might use

```
CONDICIO OCCUPATION(PERSON(DATASET):
QUE LEHRLING->1,GESELLE->2,MEISTER->3
```

resulting in a variable which contains the respective numbers for every person that has an occupation containing those words and zero for everybody else. As with the QUAERO command, logical AND, OR and NOT are available, together with a couple of further refinements.

4.3 Translations of Information Close to Numerical Values

Such translations are performed by the command CONVERSIO, operating data type dependent. For data type LINGUA it assumes that the basic information contains one or more entries following the rules given for the emerging data type NUMERUS that is, entries consisting either of numbers from the start, or of items that are numbers in some non decimal system of counting and/or measuring (ages, prices and the like).

For data type TEMPORA the internal representation of the calendar date or parts thereof are transferred, this internal representation consisting of three pairs of numbers representing the exact date of an event and its *termini post* and *ante quern*. For each of these there is given the number of days since the 1st of January of the year 1 of the Gregorian calendar and a seven or eight digit number giving year, month and day.

For data type CONDICIO, where each character is treated independently, two possibilities are provided. Think of an elementary information SEXRELFAM with the following values:

M - MALE
F - FEMALE
C - CATHOLIC
L - LUTHERAN
S - SINGLE
R – MARRIED

```
CONVERSIO SEXRELFAM(PERSON(DATASET)
```

would create 6 dummy variables indicating by a 1 that one of the six possibilities is fulfilled, by a zero that it is not. If you like regression with dummy variables less, the command

```
CONVERSIO SEXRELFAM(PERSON(DATASET):REGULA=(MF,CL,SR)
```

will probably suit you more. It would create the variables, each depicting one of the meanings contained in the original basic information. (M will be depicted as a value of 1 in the first variable, R as a value of two in variable 3 and so on).

5. Linking the Data

Thus far we have examined only the case where *one* source was converted into *one* file and analysed separately from all other sources which might be of interest for a study we are going to undertake. This approach indeed is quite often taken by historians using the computer – though mainly by reasons of efficiency, not by any inherent methodological virtue in looking separately at separated sources. Very often we would wish to connect for our analysis information contained in more than one bundle of original records – say we might wish to combine the tax returns of a series of years with a census return out of one of them, being in this way much better able to estimate the differences in family size between different social strata than by the usual way of taking the occupation of the head of household as indicator of his social position. This problem in some cases is exceptionally easy to solve, as it can happen that tax as well as census returns use the same numbering of houses. Say we have two sources with the logical structures given below.

```
STRUCTURA TAX(ROLL-NUMBER,AMOUNT-PAID,HOUSE-NR)
and
STRUCTURA CENSUS(HOUSE-NUMBER,
HEAD(SURNAME,1St-NAME,OCCUPATION,
WIFE(MAIDENNAME,1St-NAME,ORIGIN)
CHILD(SE X,1ST-NAME,BIRTHDATE)))
```

In this case we might simply merge the two sources by adding the AMOUNT-PAID – and perhaps the number in the taxroll – to the information known about the household in the census list, taking identity of house number as indicator of a desirable merger. To do so we have to specify two things for the machine

- that we want to merge two files, by using the numbering of houses as indicator
- where we want to find the added information afterwards. (In our case we will want it trivially at the very beginning – in the logical sense – of our document, as the tax is something related to the whole household, not, e.g. to any particular child. If, on the other hand we wanted to merge information about spouses out of a marriage register with the census return, we would expect every spouse to be added to the appropriate child instead of the household as a whole.)

This is accomplished by the following commands out of the CLIO command language:

```
CONNECTIO PRIMUM = CENSUSLIST,SECUNDUM=TAXLIST,
QUE NOVUM=CENSUSTAX,
QUE MODUS=MIXTURA,FONS=HOUSE-NUMBER(TAXLIST)
INCORPORATIO TOTUM(CENSUSLIST)=TOTUM(TAXLIST):
QUE NON DUPLICATAQUE
```

which mean in English:
“I want to merge some information out of the file TAXLIST with that in file CENSUSLIST, giving a new file named CENSUSTAX. The files shall be mixed that is, parts of one document shall be added to another; which tax return is to be inserted into which census return has to be decided by using the numbers of the houses given in the tax returns. The returns selected for a merger shall be inserted into the appropriate census return as follows:

- every elementary item of the first (and in our case only) group of information in the tax return shall become an item of the first group of information in the census return; if two elementary items are encountered in the two files, carrying the same mnemonic names, they shall be compared. If two are found to be equivalent, it is not necessary to duplicate the information already in the census return (so the numbers of the houses won’t appear twice).”

This rather trivial example was dealt with at such length, because it can illustrate quite well the logical structure of the system of record linkage used. CLIO provides a command language that allows independently specifying criteria for linking documents and specifying, how out of two of them a logical sum shall be computed to form the new one. As we have seen above, such commands can be extremely plain in simple cases, while the language allows the specification of very complex methods for defining identity as well as logical summing. Unfortunately a language that allows complex definitions has to provide for many things and can therefore scarcely be summed up adequately within the space one has for a general representation of a much larger system. So, rather than flooding the reader with examples of how things could be formulated, we will restrict ourselves to a general outline of the main features provided.

Apart from the case given above, where a (hopefully) unique indicator for a merger is already provided in the data, we will usually have to compare a set of information out of one file with another set out of the other to decide if the documents represented by them have to be merged. It can be extremely complicated to decide at the very beginning everything that shall be taken into consideration to do so. To minimize this problem, CLIO favors an iterative approach. Rather than thinking of everything that might indicate identity, the user is supposed to start with a very simple set of rules defining it. He will than receive a list of suggestions of which documents might be merged under these rules. He selects the appropriate ones and scans quickly through the remaining cases. Than he formulates a slightly more complex set of rules for comparison and starts the process again. After some time he will be left with a relatively small number of documents remaining, where it will probably be easier to decide about the necessary mergers out of hand, than formalizing this decision.

To smooth up this process, CLIO provides a command to create auxiliary files for such file comparisons.

```
PERSONA SURNAME(HEAD(CENSUS)->SURNAME,
QUE 1St- NAME (CHILD () ->1St-NAME,
QUE MAIDENNAME(WIFE() ->MOTHERS-NAME,
QUE BIRTHDATE(Child() ->BIRTHDATE:NOMEN=CHILDREN
```

would create one such file, named CHILDREN, providing a set of information about individuals out of the census returns we have described above. As you see, these items of information are taken out of different groups – what makes it unnecessary to consider problems of record linkage while deciding about the way you enter your data. The composite symbol "->" tells the system that, what formerly was known as maidenname of wife in the file of census returns shall in future be known as mother's name within our auxiliary file. A note of clarification might be appropriate: "Nominative Record Linkage" is a technique that has been extensively discussed during the last years within the context of family reconstitutions[16]; this being so, the examples in this paper are drawn from that field. CLIO as such has no inbuilt bias favoring Historical Demography and/or the history of the family whatsoever. Indeed some pains were taken to write the programs as general as possible. The system would therefore lend itself as easily to any other application of "linkage by nominal classification" one might think of.

Having defined two files by a series of PERSONA commands we can proceed to formulate our rules of comparison. For the linkage of a marriage register with a baptismal register, such rules might look like.

```
REGULA FATHERSNAME(BAPTISMS)=GROOM's-NAME(MARRIAGES):
QUE MODUS=DOMINANDUM, PRIMUM=MARRIAGES,
SECUNDUM=BAPTISMS, QUE CORROBORATIONES=PROPOSALS
REGULA BIRTHDATE(BAPTISMS)=GROOM's-BIRTH(MARRIAGES)+20-J
QUE :MODUS=MINOR
REGULA BIRTHDATE(BAPTISMS)=BRIDE's-BIRTH(MARRIAGES)+16-J
QUE :MODUS=MINOR
REGULA BIRTHDATE(BAPTISMS)=-GROOM's-BIRTH(MARRIAGES)+100-J
QUE :MODUS=MAIOR
REGULA BIRTHDATE(BAPTISMS)=-BRIDE's-BIRTH(MARRIAGES) +100-J
QUE :MODUS=MAIOR
```

or, translated once more into English: "Consider two cases out of the files MARRIAGES and BAPTISMS as equivalent, provided the following conditions are fulfilled:

- The name of the father in BAPTISMS is identical to the name of the groom in marriages. Consider this rule as the most important one that is, test for it first; if it is not fulfilled don't care for the other ones;

[16] A good survey: Ian Winchester, Priorities for Record linkage: A Theoretical and Practical Checklist, in: Jerome M. Clubb and Erwin K. Scheuch (Eds.), Historical Social Research (= HSF VI), Stuttgart, 1980, 114-7. Available at <http://www.ssoar.info/ssoar/handle/document/32640>.

- The birthdate in BAPTISMS has to be at least 20 years after the birthdate of the groom and at least 16 years after the birthdate of the bride. Both rules being rather rigid, consider cases for linkage, though they may violate them; just rank the probability of the linkages lower.
- The birthdate in BAPTISMS has to be not more than 100 years after the birthdate of the parents. As this rule is rather liberal, consider a violation thereof as sufficient reason to consider the link as impossible.

Write the linkage proposals onto a file named PROPOSALS."

This might be an example of the very simple rules one should use at the beginning to get rid of the many quite obvious links before bothering about more sophisticated rules for the less easily solved ones.

The result would be a printout, where for every individual in file MARRIAGES, all the information available would be printed, together with all the individuals from BAPTISMS that might belong to it, ranked according to the probability of the link. Each such proposal would be accompanied by an identification number such as 167$KW (167 being a running number, KW a pair of letters for checking purposes selected at random by the program). The user now should select the list of proposals, he wants to accept and submit the appropriate identifications to the machine twice, once that is, to perform the physical linkages, another time to get rid of all cases already decided upon out of the files BAPTISMS and MARRIAGES. It is possible to construct such rules in an incomparably more complex way; the INCORPORATIO commands mentioned provide many more facilities as well. Still, CLIO is a generalized system for the handling of historical sources, not a system for record linkages only. Let that be reason enough to restrict ourselves to two additional features of the linkage system; readers interested in further details may receive at request a copy of the relevant issue of the CLIO *Systembulletin*, documenting the progress of the system.

The most glamorous problem within the field of nominative record linkage is the comparison of nominal information, as surnames or Christian names, which are given in extremely differing spelling in the sources. Fortunately quite a few solutions to it have already been published. Usually one would therefore start to implement one version of such a solution or the other; CLIO, though, shall compare names out of sources coming from very different areas, where dialects are anything but homogeneous. So the routines doing the name comparisons were separated from the ones defining how this comparisons should actually be done. CLIO supports a concept of libraries of algorithms which may presently contain algorithms out of two classes, which can be called upon for the comparison of any given name.

```
REGULA FATHERSNAME(BAPTISMS)=
QUE SOUNDEX(GROOMS-NAME(MARRIAGES)/MYSOUNDEX)
```

would for example compare the surnames in the two files according to the SOUNDEX algorithm stored under the name of MYSOUNDEX. What such a

SOUNDEX algorithm is, has already been described quite often[17], basically it runs like this:

A name to be compared is first stripped of all prefixes and suffixes that vary most widely or are included almost at random in the sources – such as predicates of nobility, the suffix "in" designating females in German sources (*Muellerin* for *Mueller's* wife) and syllables that lend themselves to misunderstanding, their pronunciation being of no consequence within the local phonetics. Then one takes the first letter of the name pretreated in this fashion and keeps it. Now the following letters are compared with a previously defined table. If a letter has been designated to be a separator – usually vowels, letter "h" and the like – it is skipped. For the rest, one selects which phonetic group the letter belongs to and adds its serial number to the initial letter kept. This is repeated until the wanted length of the code is reached (usually three digits after the first letter). Letters of the same phonetic group appearing immediately behind each other are considered only once, unless separated by one of the separators. A typical example is the following set of rules used – among others – by the Philadelphia Social History Project[18]:

1) Retain first letter of name after prefix treatment
2) A, E, I, 0, U, W, H are bypassed.
3) B, P, F, V are coded 1.
4) C, G, J, K, S, Q, Z are coded 2.
5) D, T are coded 3.
6) L is coded 4.
7) M, N, R are coded 5.
8) Double letters are treated as single letters.

As we recognise, rules 1 and 8 were already mentioned in the general description of the algorithm. At least for CLIO they form part of all algorithms of the SOUNDEX class and need not concern the user therefore. The remaining rules would be expressed in CLIO command language by

LITTERAE SINE AEIOUWH,BPFV,CGJKSQZ,DT,L,MNR

Before we use the algorithm – as long as it is not provided by the system library – we would have to enter this directive, together with others, to define it. In our case we might decide just to strip a name of any leading "von" and of any closing "in" before we apply the rules just given. This would look like:

[17] Ian Winchester, The Linkage of Historical Record by Man and Computer: Techniques and Problems. In: Journal of Interdisciplinary History I (1970), 114-7.

[18] The project is described in a series of articles in the Historical Methods Newsletter IX (1975/1976), pp. 43-181. The record linkage system developed in its context is described there in an article by Theodore Hershberg, Alan Burnstein und Robert Dockhorn, Record Linkage, pp. 137-163. CLIO is very much indebted to some of the ideas formulated there; more so than can be seen from this paper, as quite a few ideas beyond SOUNDEX (or VIEWEX as PSHP calls one of the variants there) have been used. The algorithm taken as an example in this paper is presented there on p. 142. The record linkage article has been translated into German as Verkettung von Daten, Record Linkage am Beispiel des Philadelphia Social History Projects. In: Wilhelm H. Schröder (Ed.), Moderne Stadtgeschichte (= HSF VIII), Stuttgart 1979, 35-73. Available at <http://www.ssoar.info/ssoar/handle/document/32782>.

```
SOUNDEX NOMEN =MYSOUNDEX
LITTERAE SINE AEIOUWH,BPFV,CGJKSQZ,DT,L,MNR
PRINCIPIA
VON=
LIMITANDA
IN=
EXITUS
```

This, once more, is an extremely simple example for a feature allowing much more complex constructs; the definition of the prefix/ postfix/infix treatment of the PSHP algorithm mentioned above takes altogether 45 lines in this form. It should be further mentioned that at any time an almost unlimited number of such algorithms may be used simultaneously, as one optimal for the comparison of surnames need not be so for the comparison of Christian names or places of origin.

Besides this class of algorithms, CLIO supports another one, proposed by Gloria Guth in the Historical Methods Newsletter[19]. Using this method we compare the original names letter by letter from left to right. If two of them don't match, a number of alternatives are tested. If the 3rd letter of name 1 doesn't match the 3rd of name 2 it still might match the fourth of name 2 and so on. In its original form the method used the following rules:

	Position in Name 1	Position in Name 2
Test 1	x	x
Test 2	x	x+1
Test 3	x	x+2
Test 4	x	x-1
Test 5	x-1	x
Test 6	x+1	x
Test 7	x+2	x
Test 8	x+1	x+2
Test 9	x+2	x+2

To define a GUTH algorithm consisting of these rules you would need the following directives:

```
GUTH NOMEN=MYGUTH
PRIMUM= SECUNDUM
PRIMUM= SEDUNDUM+1
PRIMUM= SEDUNDUM+2
PRIMUM=SEDUNDUM-1
PRIMUM-1 = SECUNDUM
PRIMUM+I = SECUNDUM
PRIMUM+2=SECUNDUM
PRIMUM+1 =SECUNDUM+1
PRIMUM+2 =SECUNDUM+2
EXITUS
```

[19] Gloria J. Guth, Surname Spellings and Computerized Record Linkage. In: Historical Methods Newsletter X (1976/1977), 10-19.

As this example may prove, the CLIO directives for the definition of algorithms usually allow the user simply to write down what he has on his note pad when he finally made up his mind on how he wants his comparison done; this applies to SOUNDEX algorithms as well, though they being more complicated, to prove this would take up too much space.

6. Further Plans

As we stated at the beginning the development of CLIO started only 18 months ago. What was described in this paper is therefore not what is finally hoped for the system, but just the status quo. The lines of further development are influenced by two purposes CLIO serves. First of all, it shall support current research going on at the *Max-Planck-Institut für Geschichte* in Göttingen. Beyond that I hope to give in the not too distant future a general description of the problems related to the machine supported treatment of those types of information contained in historical sources and related to social systems of "long" duration. In this context CLIO shall prove that things asked for theoretically actually can be done.

Out of this situation the following goals for the next stage of development have grown.

1. CLIO shall in future support much better than heretofore data, which are basically numeric. This will mean the implementation of the datatype NUMERUS, already described in this paper, and several routines for aggregating data.
2. CLIO shall provide more immediate support for quantitative methodology. This will certainly include an interface into SPSS, very probably one into CLUSTAN and possibly ones into other systems.
3. On the nominal and subnominal (my name for variables, where it is not clear if two identical values mean the same thing, as often the case with the vocabulary of historical sources) levels CLIO shall offer its own analytical routines
4. CLIO will get an updating module.
5. The concept of thesauri, currently employed just for coding purposes, will be vastly generalized, allowing much more complex methods of retrieval.
6. Alternative ways to access system files will be provided, possibly allowing the introduction of a dialogue module.
7. Improved possibilities for content analysis are as well under consideration as modules supporting the reanalysis of machine readable data in formats that are logically, but not formally, close to the ones used by CLIO.
8. Of course every feature shall become faster, smaller and make the user happier.

The development of CLIO is complicated by two closely related problems. First of all: it is extremely hard to ensure the reliability of a system of such general outlay. Probably May through September of this year will be spent simply testing features with as many data as are provided by the projects in the *Max-Planck-Institut*. Still, it is clear already that at the end of this testing period quite a few bugs will remain undetected, as a whole set of options will never have been properly tested, their existence being important for a general purpose system as such, but not necessarily for the immediate needs of the supported projects within the next few months. The second problem is closely related to this one: conceptually as well as economically (it is much easier to implement a large extension of the system at one occasion than adding small improvements ad hoc) the next version of CLIO has to be designed to support the broadest possible spectrum of quantitative methods. One cannot expect that every branch of quantitative methodology will be used by the projects to be supported within the next months. The solution to both problems for professional software development would be to design special sets of data just for testing. Taking into consideration that with CLIO everything shall work for a very large number of data structures – and do so without overstraining the resources of the computer – that is simply impossible for a one man business, interested mainly in the methodological problems involved and not in the sale of software.

This situation leads to the following offer. (As usual with scientific programs the source code of CLIO is of course available free of charge. If you do not have a UNIVAC of the 1100 series at your local computing center, the implementation of the approximately 10.000 lines of PL/1 forming the current version of the system will take a rather experienced programmer. An earlier version of CLIO – without the whole record linkage modules – may be available for TR 440’s before long from Bielefeld.)

I offer the services CLIO can provide to a necessarily small number of research projects. Of course this does not mean that I would undertake any computations which can be done by software as usually available at academic computing centers. If a research project interested in such services poses problems though, which are essential for the development of a general concept of machine supported treatment of historical sources. I’m ready to integrate their solution into the development of CLIO. For practical work this would mean that you have to provide a dataset that is compatible to the CLIO formats or can be made so with small effort. Then I’m willing to build out of such datasets files that can be processed further at your local installation. Every substantial finding made during processing of such datasets belongs of course to the owner and remains inaccessible to everybody else. The “price“ for the use of CLIO is simply my right to use any methodological and/or theoretical findings made during the same stage. (Such should deal with the applicability of probabilistic measures to your material, possibilities of generalizing results based upon it and the like.)

Especially helpful – and therefore sure to be supported as generously as possible – would be projects, which deal with the following problems:

- Nominative record linkage with material that requires features scarcely tested by the projects now supported or with the intent to analyse the linked data sets with other methods. (Particularly helpful would be data someone plans to analyse as a network statistically.)
- The statistical interpretation of sources which contain measures in rapidly fluctuation systems of measurement and/or are intermingled with information which would be considered nominal. (Particularly helpful would be any attempt to analyse medieval accounting books and the like, or a longer series of the minutes of some authority of economic administration other than tax registers, or a study about source material out of the 19th century with prices in paper currency at any time of rapidly varying agio.)