ESSAYS ON BLAISE

First International Blaise Users Meeting (1992)

PROCEEDINGS OF THE FIRST INTERNATIONAL BLAISE USERS MEETING

14 - 16 October 1992
Netherlands Central Bureau of Statistics, Voorburg

NETHERLANDS CENTRAL BUREAU OF STATISTICS
Statistical Informatics Unit
P.O. Box 959, 2270 AZ Voorburg

September 1992 BPA nr: 10892-92-M3

PREFACE

The papers assembled in this proceedings were presented at the First International Blaise Users Meeting, which was held from 14 to 16 October 1992 at the Netherlands Central Bureau of Statistics. The objective of this meeting was to provide a platform for exchanging experiences with the use of Blaise in survey data processing, to meet Blaise users and members of the Blaise development team, and to discuss current and future developments.

The proceedings contain two types of papers. In the first place there are papers in which Blaise users describe experiences, problems and solutions. In the second place, there are papers by the Blaise team that cover new developments, or stress special aspects of the present version of Blaise and related software.

The programme committee consisted of Jelke Bethlehem, Maarten Schuerhoff and Sandra Vogelesang. The editing of the proceedings was carried out by Sandra Vogelesang. The meeting was sponsored by the Netherlands Central Bureau of Statistics.

CONTENTS

ELECTRONIC PUBLICATION OF STATISTICAL DATA

Alex van Buitenen, Anco Hundepool and Aad van de Wetering
Netherlands Central Bureau of Statistics, Voorburg, The Netherlands

1. Introduction

Statistical offices collect, process and produce large amounts of statistical information. The final step in this process is the dissemination of the collected statistical information. Traditionally this is done by publishing a vast number of books and periodicals. Everything is of course published on paper. As (micro-)computers are now a common tool in almost every office, there is a need to reconsider the way statistical offices should publish information.

As almost all statistical information users have a computer at their disposal, they want to use the statistical information on their computer. Of course they do not like the idea of having to key in the figures from the statistical publications into their machine. They want the information in a machine readable way. The statistical offices have to become aware of new publishing media for their information and as a consequence have to offer the appropriate services.

Besides information traditionally published in books and periodicals, there is a growing demand from researchers for the release of more detailed (if not individual) data. As the packages to perform statistical analysis are widely available also on micro-computers these researchers want to do their own analysis on data files with individual data. This leads to a growing demand for these files with individual data.

Of course, there is a great risk of disclosure in releasing files with individual information, but that is not the issue of this paper. See Bethlehem et al. (1990) for this issue. Once a statistical office is willing to release files with individual information to researchers, they are obliged to supply also the necessary meta-information to enable the researcher to understand the meaning of the different variables.

In the remaining sections of this paper we will discuss how we think that individual data files as well as more aggregated data should be made available to the public and how we should supply the necessary meta-data information to the user.

2. Aggregated data

Most of the data published by statistical offices are aggregated figures. These figures are the result of a long production process; beginning at the collection of the data, editing the data, if necessary weighting the data for the sampling method used and for nonresponse and finally, tabulation. These tables will then form the main part of most statistical publications.

As the need for electronic publications is growing, the Netherlands Central Bureau of Statistics has decided to meet these demands. The most simple and most easy way would have been to copy some data files or tables on a diskette and make them available to the users of statistics. This is however not a very friendly way of publishing data. At least the user needs to know what is on the diskettes. What is the meaning of all the variables, what are the codes used etc. This meta-data is almost as important as the data itself.

Therefore, it was decided to develop a general program (CBSview) which would meet these needs. With CBSview it is possible to publish the information together with the necessary meta information. CBSview is a general shell to publish statistical information and can be used for all kinds of publications. CBSview makes it possible for the users of the statistical information to make their own selections from the data, consult the information and to export the selected information from CBSview to almost any statistical package for further processing.

At all stages of the selection of the variables, the user gets the information he needs to make the selection, such as the description of that variable. After having made the selection, the user can specify the format he wants for the selected information. At this moment we support the following formats:

- Tables in ASCII format. This kind of output is always generated, and can be saved on disk for later use. E.g. these tables can be included in your own reports.

- A Lotus 1-2-3 worksheet. This direct link to Lotus and also Quattro is meant for those applications, where the user wants to make his own calculations or wants to use the large graphical capacities of these programs. The current version of CBSview does not have any graphical possibilities nor can it perform calculations. This is left to the spreadsheet package.

- A dBase file. This file can also be read by other database packages like Paradox.

- An ASCII data file. This general file can be used to read the selected data into various statistical packages like SPSS and SAS.

- Setups to read the ASCII file into other packages. CBSview is able to create setups for SPSS, Stata, SAS, Abacus and Manipula. The latter two packages originate from the Netherlands Central Bureau of Statistics and are meant for tabulation and file manipulation.

- A file containing the descriptions of the selected variables.

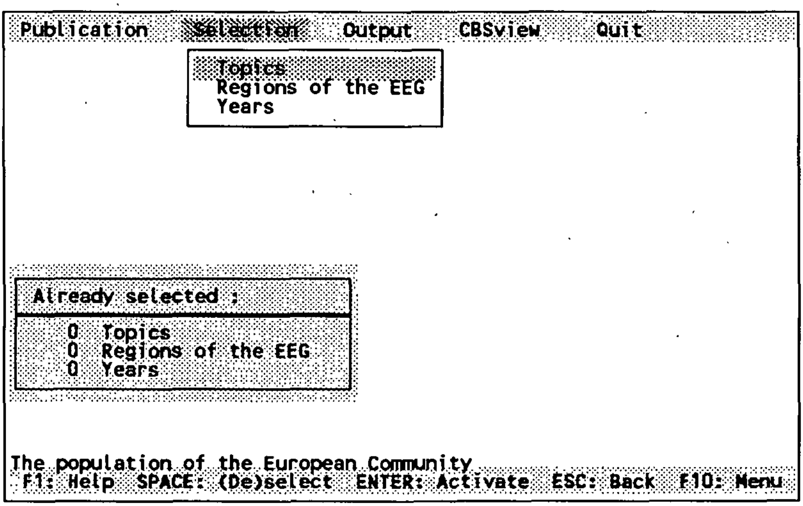
At this moment the structure of the information that can be stored in CBSview is a homogeneous data matrix (up to 10 dimensions). The user can select parts of this data matrix by selecting variables in each dimension.

Although this format is suitable for most publications, it has proved to be too restrictive for other publications. For these other publications the basic information consists of a (large) set of small tables. Many statistical publications are a compilation of a set of tables. In version 2 of CBSview it will be possible to put these kind of publications on diskette. This will make it possible to inspect the basic tables on the screen, export them to a file to include them in reports and convert the tables to a Lotus 1-2-3 worksheet.

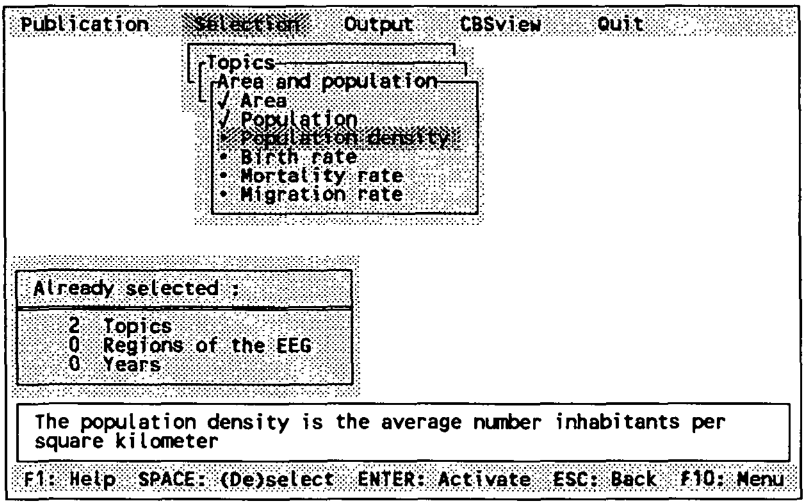
This version of CBSview will also include a thesaurus to help the users to locate the information they are interested in.

3. The use of CBSview

The use of CBSview can be divided into two stages. In the first stage, after the publication has been chosen, the user makes a selection of the data available in the publication. A menu driven program helps the user to select the information he is interested in. As the information in a CBSview publication is a homogeneous (more dimensional) data matrix the user must select items from each dimension. In the following screen the user chooses one of the (in this example) three dimensions, where he wants to make a selection. The first dimension is always called topics (variables), while the names of the other dimensions can vary depending of the publication. For example regions for a regional publication or years for time series.



After selecting the item 'Topics' the user will access a hierarchical tree structure in which all the items are structured. This tree structure can be several levels deep. The menu program leads the user through the hierarchical tree structure. At any level the information about the current item is displayed in a box at the bottom of the screen. Also some information about the number of selections made is displayed on the screen.



In a similar way the user makes his choices from the other dimensions. If the selection of the information is completed the user moves to the next step, i.e. the actual retrieval of the selected information. In the following screen the user can choose the kind of output he wants to be generated. More than one choice is possible.

```
Publication   Selection   Output   CBSview   Quit
                          Define export files
                           Filename    :  EEGVIEW
                           Output type       | Y/N | Extension
                           Screen            |  Y  | SCR
                           Ascii             |  Y  | ASC
                           Rec.descr         |  Y  | ASR
                           Explanations      |  N  | TLT
                           Lotus             |  Y  | WKS
 Already selected :        Dbase             |  N  | DBF
                           ABACUS-setup      |  N  | ABC
                           MANIPULA-setup    |  N  | MAN
   4  Topics               SPSS-setup        |  Y  | SPS
   3  Regions of the EEG   STATA-setup       |  N  | DCT + DO
   1  Years                SAS-setup         |  N  | SAS

The population of the European Community
 F1: Help                                     ESC: Back  F10: Menu
```

Finally the user instructs the program to do the retrieval and the selected information will be presented on the screen as the following table. If the table is too big to fit on a screen the user can scroll through the table.

Source: EUROSTAT Region Years	Area and population			
	Area	People	Density	Birth
	Km²		Per km²	per 1000
Netherlands 1986............	41 509	14 572	351	12.7
North Netherlands 1986............	9 078	1 591	175	12.4
East Netherlands 1986............	11 304	2 949	261	13.3

```
 The population density is the average number of inhabitants per
 square kilometer
                                                       ESC: Stop
Scroll:↓,↑ [CTRL] ←,→,PGUP,PGDN HOME,END ESC:Stop F9:Info F10:Menu
```

4. Individual data

A different approach is followed for individual data. In this situation the user in general does not want to see the individual records of the data file, but either he wants to generate his own tables from the data or he wants to make his own analysis of the data. But here too the availability of good meta information is very important. When publishing individual data there is great risk of disclosure of individual information. At the CBS there is a research project running to study these problems. As a result of the study a prototype of a program ARGUS (De Jong et al., 1992) has been developed, which helps to identify the possible disclosure risks. The problems of disclosure however are not the subject of this paper.

When we publish a data file with individual records, we assume that the data file has been protected against disclosure and therefore can be made available to the public. Here again we do not want to make a simple ASCII file and leave it to the users to work it out. The users need to know at least what the meaning of the variables is, what the coding of the categories stands for, etc.

We have chosen to publish these files in an ASCII format. Not because we think that everyone wants to use this ASCII file, but it is a very general format that can be read by various programs. Besides this ASCII file with the data the user needs the meta information. Therefore, we supply a system which is capable to converting the meta information into a format useful to the user. This format can be a plain record description but also a setup for transforming the data into the format of the various analysis packages like SPSS, SAS and Stata.

5. Publication media

Up to now we have only discussed the publication of information on diskette. Other computer media are available, of which the CD-ROM is going to play an important role in publishing statistical information . The CD-ROM has a very large capacity (about 500 Megabytes) and is also very reliable. Once the data have been written on CD-ROM they are very secure. Although CD-ROM is a slow medium compared with a hard disk, it will be (and is) used for publishing statistical information. With respect to the software (CBSview) using a CD-ROM to publish information is not a source of great problems. However in the Netherlands we still wait to start the use of CD-ROM because of the relative small availability of CD-ROM players in the Netherlands. Also the capacity of a CD-ROM is a point. We do not have a census any more and therefore we do not have those very large data files to justify the use of a CD-ROM.

An other possibility is the use of a publication computer (server) located at the statistical office. In this situation the data files to be published are on this computer which can be accessed by the use of a modem and a micro computer. The same software (CBSview) can be used to make the information available. In this situation the user runs on his own computer the specification part of CBSview (the 'front end') and only accesses the central computer (the 'back end') to read the selected information from the CBSview databases. This way of working implies that the specification part of the software is made available to the users to run it on their own computer. They download the meta information of a publication only once and use it (off line) to specify a selection.

6. Conclusion

It is very important for the statistical offices to be aware of the changing demand for information from the public. The computer will play (and is already playing) an important role in the dissemination of statistical information. The use of adequate software to disseminate the statistical information facilitates the use of this information. It should be the goal of a statistical office to achieve that the information it has gathered is used in as many places as possible. The use of the statistics produced is the main reason why a statistical offices exists.

The Netherlands Central Bureau of Statistics has developed software (CBSview) to disseminate statistical information on electronic media. The first reactions from the public are very encouraging. Although the use of

CBSview is restricted to CBS publications, the same software is available (as STATview) for other agencies to make their own publications.

References

Bethlehem, J.G., Keller W.J., Pannekoek J.(1990): "Disclosure Control of Microdata", *Journal of the American Statistical Association* Vol.85, No. 409, p 38-45.

Jong W.A.M. de, Willenborg L.C.R.J., 1992, "Argus: an integrated system for data protection", *Proceedings of the International Seminar on Statistical Confidentiality*, Eurostat/ISI.

FINNISH NATIONAL HEALTH INTERVIEW SURVEY USING BLAISE CATI

Timo Byckling
Statistics Finland, Helsinki, Finland

1. Conversion from personal interviews to CATI

From the early sixties onwards a series of nationwide health surveys have been carried out in Finland to monitor the health status, the use of health services and other important health-related aspects of the population (Social Insurance Institution 1964, 1968, 1976, and 1987). These studies have required a great deal of resources; they are based on large samples (in the order of 15.000 persons), trained nurses work as interviewers, and the personal interviews take one and a half hours on average to complete. Even in better economic times such expensive large-scale health surveys based on personal interviewing are feasible only every 5-10 years, and the publication of the main findings lays 2-3 years after the field work.

The CATI collection method seems to offer a fast and cost-effective instrument for an annually executed health survey covering all the main health status and health services indicators for which data exist also in the previous health surveys. Reduced costs were a strong argument in favor of the new data capture approach. The period of economically difficult adjustments through which Finland is now going, and which is reflected in the changing utilization patterns of health services, calls for a relatively inexpensive survey run on a yearly basis. A well-executed CATI data collection system would seem to be suitable for the monitoring of the rapid shifts in the use of health services.

Several reasons speak for a CATI application in health services research:
(1) In Finland 93% of the households have access to telephone; therefore, undercoverage due to non-telephone households is manageable (however, unlisted numbers plus the difficulties of matching sample persons with respective household telephone numbers present complications).

(2) The usual arguments for the adoption of CATI techniques, timeliness, improved data quality, reduced costs, and sample management by the CATI program apply equally to the collection of health survey data.
(3) Blaise CATI offers several facilities, which enable the researcher to do things he could not do with the Paper&Pencil method: On-line automatic coding of diseases, use of the computer's clock for fixing reference periods for retrospective questions on health sercives use, and internal consistency checks on services utilization questions.

2. Finnish Health CATI

The first Finnish "Health CATI" was undertaken in the fall of 1991 and will be repeated yearly. Statistics Finland which is responsible for the data-collection phase of the research process has a CATI unit in Helsinki with 10 microcomputers in a Token Ring/Novell based network. In this experimental stage of the institution's CATI activities volunteers from the interviewing personnel of Statistics Finland were recruited for the health study.

Data were collected on perceived health, long-standing illnesses, and the use of health services (medical, preventive and dental services, physiotherapy, occupational health services, medicines). In addition, opinions on health care fees and the coverage of personal doctor services were asked.

A total of 2274 persons (aged 25-79) were interviewed in 1991 which represents 72% of the original sample. 15% of the sample persons either had no household telephone number, had an unlisted number or had a number which could not be found while matching sample information with a telephone directory. Refusal was the outcome in 6% of the cases, and 7% of the sample could not be reached by phone during the six weeks of field work. The participation rate in this survey is approximately 10 percentage points below the level for personal interview surveys conducted by Statistics Finland. Methodological work is underway to study in more detail the effects the non-response has on survey estimates. Participation varies according to age, household size, sex and geographical location.

The sample persons were notified beforehand by mail about the upcoming interview, which helps considerably the initial contact; also a good portion of the non-telephone group send their telephone numbers to our office.

Average interviewing time was 16 minutes; the interviewers spent more time on the phone with older people who have more health-related episodes, but senior citizens are also more interested in telling about their illnesses. The interviewers had no difficulties whatsoever to establish a good rapport over the phone with the respondents, which was of some concern to us at the outset of the study, since a good rapport is vital for the success of a health survey and earlier health interviews in Finland had used a different approach to data collection: trained nurses made face-to-face interviews in people's homes. According to our observations, health status indicators are sensitive to the mode of collecting information about chronic illnesses or symptoms, and methodological work is needed to sort out these differences in more detail. Comparing personal interview data with CATI results has to be done with great care.

3. Difficulties in reaching people by phone

In telephone surveys, in contrast to personal interviews, no possibility exists to come up with additional information which would help in contacting the target person (e.g., consulting neighbours). If the outcome is "no answer" after several dials, we are groping in the dark as to how to contact the target person. In Finland we can use population register information, such as age, occupation and geographical location to improve the probability of contact. Optimal call scheduling based on call history should be developed, as well.

One-person households pose special problems in this respect. The following table gives the participation rates in the 1991 Health CATI by sex, age and the household composition (one-person vs. other households):

Women							
	Age	25-44		45-64		65+	
	Household size	1 p.	2+	1 p.	2+	1 p.	2+
	Participation (%)	45	71	72	79	79	82
Men							
	Age	25-44		45-64		65+	
	Household size	1 p.	2+	1 p.	2+	1 p.	2+
	Participation (%)	49	65	47	80	54	86

The figures in the table show clearly that one-person households pose a real problem in a CATI study. For instance, living in a one-person household pushes the participation rate down 12 percentage points among men. Age is also important in determining the probability of successful outcome.

4. How many re-dials are needed?

Possibilities to focuss on "no answer" cases would increase, if we had a better way to control the scheduling of re-dials in the Blaise program. Let's say we have decided to have up to eight re-dials per target person. If we get two rings with no answer on two successive afternoons, we might want to dial next time in the morning, and if necessary, postpone the remaining dials until a week or two from the last unsuccessful ones. In the course of time optimal call scheduling strategies should be devised for different population groups.

In the Finnish study a good many dials were made for experimental purposes before giving up a case (too many dials for cost reasons!). 95.5% of the interviewed sample was reached with at most five dials; a maximum of nine dials raised this figure up to 99.5%, and the remaining half percent required over nine dials. Thus, it seems that in the Finnish population a minimum of five dials is recommended, but this number could be raised to 6-10 dials depending on time and cost constraints.

5. Training for Blaise CATI

The Blaise program functioned fully to the interviewers' and the researcher's satisfaction in the Finnish Health CATI study. For the questionnaire designer Blaise's facility for producing neat-looking paper questionnaires is very valuable; for instance, in our CATI survey we went through five versions in an expert group before reaching the final one.

Although the interviewer can see only one question at a time in the upper part of the screen, the lower part gives her a good enough overview of the context in which the particular question is embedded (it is a good idea to have descriptive question names to give a hint about the question contents).

According to our experiences, training regarding the Blaise program should include the following points:

(1) The interviewer should be comfortable in using key combinations, erasing and making changes in responses, and going through the questionnaire, especially in skipping back to check earlier answers.

(2) The CATI questionnaire should be designed such that it is "transparent" from the point of view of the interviewer because her main responsibility is good interviewer performance. The interviewer must trust the CATI instrument and be proficient in its use but apart from this, she should pay only marginal attention to the instrument itself. Since computer-aided interviewing programs tend to segment questionnaires in small fragments (although Blaise's design is better in this respect than that of most other programs), the questionnaires should be designed keeping the interviewer's work in mind; for instance, it is imperative that when a study question appears on the screen the interviewer is able to comprehend it quickly and that the screen provides the interviewer with all the necessary information. Blaise would certainly serve the interviewer better if she had easy access to context-sensitive help screens giving supplementary background information about survey items. Our experience is that when problem situations occur in the course of the interview, the respondents do not mind delays if they know why they occur, but the interviewers themselves experience these situations as embarrassing.

(3) The Blaise calendar is a useful tool for making appointments, but the interviewing staff needs to be trained to use it properly. Messages that our interviewers wrote to the persons who picked up next an unresolved case left much to desire. A smooth management of a case in a CATI unit presupposes that the interviewer when making an appointment in the calendar gives exact information about the date and time of the contact, who was on the phone, what was agreed upon and all other useful information concerning the case. Then the next interviewer can easily pick up the case from the point where it was left during the previous contact.

A small unit like ours does not need a full-time supervisor to keep an eye on interviewer work. An experienced interviewer with enough training in Blaise and network functions has been trusted with the care of the day-to-day chores (running the day batch, overseeing interviewer performance, taking backups, bringing problems to the researcher's attention etc.). One researcher carries a cellular phone with him at all times (I once got a call from the CATI center while I was mixing with people in an art exhibition!).

THE DEVELOPMENT OF THE FAMILY RESOURCES SURVEY IN BLAISE FOR THE DEPARTMENT OF SOCIAL SECURITY, G.B.

Paddy Costigan, Sven Sjödin, Social and Community Planning Research, London, and Katarina Thomson, Joint Centre for Survey Methods, London

1. Introduction

This paper is based largely on the experience of Social and Community Planning (SCPR) in developing a Blaise questionnaire for a large and complex continuous survey, and on earlier work undertaken at SCPR and the Joint Centre for Survey Methods (JCSM) [1].

The start-up costs of acquiring expertise and laptops mean that simple surveys alone are unlikely to justify the expense of Computer Assisted Personal Interviewing (CAPI) in the near future despite the merits of 'clean data straight from the field'. Hence CAPI needs to prove its benefits for more complex surveys where extra constraints may come into play, such as hardware and software limitations, data storage, and the efficiency and maintainability of the program [2].

---

1) J. Martin and C. O'Muircheartaigh *Evaluation of Computer Assisted Survey Systems, Report 1: Introduction to Computer Assisted Survey Systems*, JCSM Working Paper Number 4, 1991.

2) A paper on this topic is also due to be delivered to the conference on Survey and Statistical Computing organised by the Study Group on Computers in Survey Analysis. The proceedings of that conference will be published in A. Westlake et al *Proceedings of the conference on Survey and Statistical Computing, Bristol, September, 24-26 1992*, Elsevier Science Publishers B.V., Amsterdam (forthcoming).

2. Key features and requirements of FRS

The Family Resources Survey (FRS) [3] is a complex survey: it is large and continuous (25,000 households per year); it deals with several hierarchical levels of data; it is a long interview (average 90 minutes per household); the co-operation of all adults in the household is required for the interview to be considered fully productive; and the interview collects large amounts of very detailed financial information, wherever possible verified by reference to source documents either in the interview or later.

3. Evaluation and choice of CAPI software

SCPR carried out an evaluation on behalf of the DSS of three leading CAPI software packages for potential use on the FRS. This was done by bench-testing them against a list of evaluation criteria. Blaise was chosen as the only package with sufficient potential for the FRS, principally because of the ease of movement around the questionnaire and the ability (using the *Table* facility) to display simultaneously, and move between, the answers of two adults being interviewed concurrently. There were, however, concerns over Blaise's capacity to deal with large, hierarchically structured questionnaires. Section 4.1 below discusses the solutions adopted.

4. Programming the questions

This section describes the problems encountered and solutions adopted in dealing with a number of major programming issues.

---

[3] The FRS is sponsored on behalf of the UK government by the Department of Social Security (DSS). The survey is due to be launched in October 1992, and will be carried out jointly by SCPR and the Office of Population Censuses and Surveys (OPCS).

4.1. Data hierarchies

The FRS contains complex data hierarchies: each household may contain one or several Benefit Units [4]; each Benefit Unit may contain one or two adults plus children; and each person (adult or child) may have several items of a particular sort, where information needs to be collected for each item, e.g. jobs, occupational pensions, financial holdings.

Because of the restrictions on using loops within loops in the ROUTE paragraph in Blaise version 2.4, it can be rather difficult to implement multiple levels of hierarchy. E.g. it is **not** possible to use the following pseudo-code to ask about three occupational pensions for each of ten persons in the household [5]:

```
FOR i := 1 TO 10 DO
   FOR j := 1 TO 3 DO
      Pension[i,j];
   ENDDO;
ENDDO;
```

Two levels of hierarchy can be created by using a loop within a table, but tables are not always the most suitable format. Two levels of hierarchy can also be created by defining questions as a block and declaring the block several times, but this is really only suitable if the inner loop is very small, e.g.:

4) A Benefit Unit generally consists of either a single person (with any dependent children) or a married or cohabiting couple (with any dependent children).

5) For clarity's sake, the question of whether there actually are ten people and whether they all have three pensions has been ignored in this example.

```
BLOCK PensionQuestions;
  .......
ENDBLOCK;

QUEST
  Pension1 : PensionQuestions;
  Pension2 : PensionQuestions;
  Pension3 : PensionQuestions;

ROUTE
  FOR i := 1 TO 10 DO
     Pension1;
     Pension2;
     Pension3;
  ENDDO;
```

By using tables, loops and re-declared questions in combination it is thus possible to create three levels of hierarchy. This is therefore not sufficient to deal with the FRS levels of Household-Benefit Unit-Person-Pension. A different approach was needed.

A related problem is run-time data storage space. The survey procedures allow for up to ten adults per household: the above design would require space for 30 instances of pensions for every household, whereas most households would probably only have one or two.

It soon became obvious that the questionnaire was in any case too large to be contained in one program. The original paper and pencil (PAPI) design had consisted of two questionnaires - one for the household and one for multiple adults. This was therefore a natural place to split the program.

A by-product of splitting the questionnaire was that the hierarchy problem was reduced. Thus the household program is run once for each household; a Benefit Unit program is run once for each Benefit Unit, with concurrent interviewing of its adult members. Up to nine Benefit Unit questionnaires can be linked to the household questionnaire. Thus within the Benefit Unit program, the upper layers of hierarchy can be ignored.

So, in the Benefit Unit program, the first level of hierarchy is whether you are dealing with Adult 1 or Adult 2 within the Benefit Unit. This is mostly handled via the table structure (see section 4.2 below on concurrent interviewing). The next level (e.g. multiple pensions) can be implemented using either a loop or a multiple declaration of questions, whichever is easier in the context (see section 4.3 below on different ways of implementing repeated questions). It would even be possible to implement a further level of hierarchy (e.g. some item of which there might be several per pension) by a combination of a loop and multiple declarations of questions.

This approach also reduces the data storage allocated: e.g. storage space for six pensions (i.e. three pensions each for two persons) is now required for all households. This is increased only if there are several Benefit Units in the household. This is clearly a substantial improvement on the figure of 30 given above.

The linking of the household and the individual programs is achieved via hierarchical serial numbers. The household program serial number contains both an address serial number and a household serial number (implemented as separate KEY questions) [6]. The Benefit Unit program has an additional KEY question: benefit unit serial number. Blaise reads information from the household questionnaire via an External Paragraph to several corresponding Benefit Unit questionnaires.

There are, of course, disadvantages in splitting the questionnaire into several programs. When the interviewer finishes one program, (s)he has to exit to an outer menu (currently implemented in Clipper) and then descend again for the new program. This becomes a particular problem when checks span questions in several programs. It is easy enough to implement such checks using information from the external paragraph, but they cannot take advantage of the usual Blaise facility whereby you can return to any question involved in the check simply by highlighting

---

6) The sample is address-based and the sample-design must allow for multi-household addresses. The highest level of data analysis is, however, the household.

the question name. If the interviewer finds that (s)he has to alter an answer in an earlier program, (s)he has to swop programs and manually find the offending question. Also, structure checks to ensure that all relevant Benefit Unit questionnaires have been completed have to be implemented outside Blaise.

4.2. Concurrent interviewing

It was a requirement of the FRS that it should be possible to interview several adults concurrently. Ideally, this should be possible for all adults in the household, but in practice it has been limited to the two adults in each Benefit Unit.

There are several reasons for the requirement for concurrent interviewing. First, a number of questions deal with financial resources or income which may be held jointly by the adults in a Benefit Unit, e.g. savings accounts or the receipt of social security benefits. There is less risk of double-counting if both adults are interviewed jointly and checks can be programmed to prevent the double-claiming of benefits. Secondly, the interview is already very long (90 minutes per household). Concurrent interviewing involves some time saving (e.g. as some questions are read out only once). Thirdly, for the interview to be regarded as productive, all adults in the household must co-operate. There is less risk of 'losing' one respondent, if they are all interviewed together.

The model for the concurrent interviewing on the FRS is the Family Expenditure Survey, which has been run for many years by OPCS. In PAPI, the questions are printed down the left hand side of the page, with multiple answer columns for the adults. The interviewer has full discretion over the question order: (s)he can choose to ask each question of each adult in turn, or (s)he can ask a few questions of one person and then turn to the next person. The latter might be more sensible, say, if there is a series of questions all dealing with entries on the person's salary slip. If a person drops out, either temporarily or for the rest of the session, the interviewer can mark the spot and carry on with the other respondent(s). This freedom has clear advantages in terms of interview dynamics, but leaves the door wide open for routing errors, especially as the FRS incorporates very complex routing.

The minimum requirements for the concurrent interviewing in CAPI were felt to be:

(a) that it should be possible to interview two persons (i.e. one Benefit Unit) at the same time;
(b) that it should be possible to run the interview with only one of the persons present; and
(c) that it should be possible to start the interview with two persons present, allow one person to drop out and continue the interview with the other person.

The first requirement - the ability to interview two persons at the same time - has been implemented using the *Table* structure. The Benefit Unit program thus consists mainly of a series of tables, each with two rows (one for each person), where the questions appear as columns.

In CAPI, the question order and where to switch between respondents has to be predetermined by the program designer. Where a sequence of questions seems to go together naturally, they have been put in the same table. Considerable pilot work has been done to establish how long to make the tables and where to place the breaks.

As for requirements (b) and (c), our approach - after a number of false starts - has been to intersperse the program with "Do you want to continue with Person x?" questions. If a person is 'suspended' at such a question, no further questions appear on the route for this person. The questions for the other person in the Benefit Unit are unaffected. If a respondent is not present at the start of the session, they can simply be 'suspended' at the outset and will be ignored for the rest of the session. If they drop out half way through, the interviewer will try to persuade them to stay until the next 'Suspend' question. If they insist on leaving between two 'Suspend' questions, the interviewer answers their questions using the 'Missing Information' key (see section 6.5) until the next 'Suspend' question. If a person reappears, and the interviewer wants to 'catch up' with them, (s)he goes back to the question where they were first 'suspended' and changes it to 'continue'. Their questions will then reappear on the route. Once coded as 'continue', the answer to the a 'Suspend' question is protected, so the respondent's answers to subsequent questions remain displayed in all circumstances and their data cannot be lost.

4.3. Alternative strategies for repeated sets of questions

There are numerous places where the same questions are asked once for each of a number of items or persons. We would identify three main ways of presenting such sets of repeated questions. For example, the household questionnaire includes a number of questions about members of the household who need special help or attention. This could be presented in any of the following ways:

(a) An initial question finds out how many people in the household need special help. The program then repeats the sequence of subsidiary questions the requisite number of times. The sequence has to start with a question along the lines of "Who is the (first/second/third) person in the household who needs special help or attention?". (The word 'first', 'second', 'third' etc. can be inserted using text substitution.)

(b) An initial question finds out whether anyone in the household needs special help. If the answer is yes, the program begins the first iteration of the subsidiary questions. The sequence again has to start with a question along the lines of "Who is the (first/next) person in the household who needs special help or attention?". Either that question has an answer option 'No one else', or the sequence has to end with a question along the lines of "Is there anyone else in the household who needs special help or attention?".

(c) An initial multi-coded question establishes who, if anyone, needs special help. If one or more persons are coded at the multi-coded question, the subsidiary sequence is repeated for each person affected. There is no need to ask each time who the person is (the correct name can be substituted into the question text) and there is no need to ask at the end of the sequence whether anyone else requires special help.

Our view is that, in general, (c) is the most elegant solution and the one that is easiest for the interviewers to deal with. Methods (a) and (b) could lead to confusion if there are a large number of items, and interviewers may end up duplicating or missing out information.

There are, however, situations where (a) or (b) might be preferable. When asking, for example, about motor vehicles available to the household, it seems more natural to use approach (a) and start by asking how many motor vehicles are involved. It is also perfectly natural to ask for each motor vehicle to whom it belongs.

Another consideration is data storage. If the program allows for up to ten adults and up to three cars per adult, storage would have to be allocated for 30 possible cars. If approach (b) or (c) was used, it might be thought sufficient to allow for a maximum of ten cars per household.

Finally, we have found it useful to implement some of the loops described above in the table format, so that the interviewer has an overview of the information given. This is particularly the case where the subsidiary sequence of questions is quite short and fits into one screen. If the sequence is longer (or contains large fields like open questions) so that the table scrolls sideways, the benefits of using a table are less apparent.

5. Checks and edits

One of the strengths of CAPI is that many checks normally done at the post-fieldwork edit stage can instead be done during the interview. Range errors can thus be eliminated and implausible information checked directly with the respondent. This improves data quality and reduces the editing time after fieldwork.

However, these benefits do not flow automatically from CAPI: they have to be worked at during the design stage. CAPI also introduces new checking requirements arising from the additional roles of the interviewer as key-punch operator and 'broker' of edit failures.

5.1. Strategies for minimising keying errors

Once interviewers have entered the data and it has been accepted, there is no independent record against which to check the answers. One hundred per cent verification of entries, i.e. the typing of each answer twice, can be used for a few critical questions, but is mostly impractical. For single-coded questions, the display of the short answer name next to the answer field is helpful in trying to cut keying errors. However, this does not operate within tables or in multi-coded or numeric questions. A new type of check is therefore needed to guard against interviewer keying errors.

For numeric data, soft checks on keying errors should be designed to be triggered whenever there is a real risk that the interviewer has keyed a zero too few or too many or has misplaced the decimal point, for example, to catch the interviewer who enters 500000 or 5000 instead of 50000. In some cases, it may be necessary to have fairly detailed information about the distribution of variables within the general population and important subgroups to design realistic checks. Non-numeric questions may also benefit from soft plausibility checks to catch both respondent and interviewer errors.

5.2. The design and storage of error messages

As discussed below, checks should be designed so as to be triggered only on rare occasions. One consequence is that the error messages will only rarely be seen. It is therefore essential that error messages are self-explanatory. The researcher needs carefully to think through and specify all error messages. It is important to remember that an 'error' can arise in at least three different ways: the respondent makes an error in answering;

the interviewer makes an error in keying the answer; or the respondent genuinely has circumstances which were not foreseen by the questionnaire designer. Error messages need to alert the interviewer to the problem and suggest reasons why it may have arisen without causing the interviewer to confront the respondent with the problem in an antagonistic way.

For some questions it is useful to store error messages as global text variables. This is appropriate when the same message applies to several checks, as it facilitates program maintenance and saves run-time storage space.

5.3. Practical limits on the number and types of checks

A necessary, but not sufficient, condition for 'clean data straight from the field' is that run-time edit checks can replace the entire office edit stage of a survey. We have already seen that CAPI introduces new requirements for checking. Checking requirements are therefore likely to be very heavy. There are, however, practical limits on the numbers and types of checks that can be implemented at run-time.

First, if the program triggers a hard check when there is no real error, the interview will come to a grinding halt. If the program frequently triggers soft checks when there is no real error, the interview becomes unwieldy and interviewers may start to ignore the error messages.

Secondly, checks implemented at run-time can really only deal with issues where the respondent can be expected to have the information. It is workable to implement a check that the amount received per week in Child Benefit should be made up of e.g. £9.65 plus multiples of £7.80, but not to implement a similar check on the amount of some benefit where entitlements vary depending on circumstances, and respondent and interviewer are unlikely to know the detailed eligibility rules.

Thirdly, there are situations where the run-time check has to be soft but clean data is nevertheless required, so a hard check needs to be imposed in the office. For example, a run-time check that only widows receive Widow's Benefit would have to be soft - you cannot take the risk of the interview grinding to a halt if the respondent insists that reality is otherwise. But it might be necessary to clean the data afterwards and enforce that only widows can receive Widow's Benefit.

Fourthly, we have found extensive checks to be costly in terms of program size and programming time.

Our experience is therefore that an office edit stage may still be necessary on a complex CAPI survey, although it should obviously be less time-consuming that the edit stage on a PAPI survey.

6. Significant limitations of Blaise version 2.4β

This section discusses some significant limitations that we have found in Blaise versions 2.37 and 2.4β. Although the limitations are often annoying and have sometimes constrained the way in which we have designed the program, none of them has proved to be an insuperable problem in the implementation of the FRS using Blaise.

6.1. Limitations on the depths of data hierarchies

The limitations imposed by Blaise on the depths of data hierarchies are potentially very serious for some surveys. The approaches used on the FRS to by-pass these limitations are discussed in section 4.1 above.

6.2. Size constraints

Even after splitting the FRS questionnaire into two programs (see section 4.1 above), we experienced a series of size constraints when working in Blaise version 2.37, mainly at compile and run time. These constraints were so serious that we were at one point forced to split the questionnaire into three programs. However, the arrival of Blaise version 2.4β and the installation of DOS version 5 (which uses less RAM) on the laptops has alleviated these problems and enabled us to return to the two-way split of the questionnaire.

A lesser, but nevertheless annoying, size constraint is that the Blaise syntax checker is prepared to allow the assignment of much longer strings than the Turbo Pascal compiler. If very long strings have to be assigned, this should be done in two stages, e.g.

```
COMPUTE LongStr := 'This is the sort of long string that the Turbo
Pascal compiler ';
COMPUTE LongStr := LongStr + 'is not going to like at all. That is
so annoying!';
```

6.3. Numeric questions and variables

Although we were generally happy with the ability of Blaise to handle numeric data, we have found a few problems.

6.3.1. Rounding errors with real numbers

It is common practice in computing to recommend that equalities should never be used with real numbers, because of rounding errors when the numbers are stored in base 2 in the underlying hardware. Since this problem is common to all programming, it is not surprising that it should surface in Blaise. However, since Blaise is specifically designed to be user-friendly to non-programmers, the documentation should perhaps carry a warning about this.

Moreover, there is a particular problem where real numbers are used to define the range for a numeric question. If you define a range to be 0.00..99999.97, you would expect it to be possible to enter the number 99999.97. But this is not necessarily the case. Assigning the range 0.00..99999.00 to the question solves the problem.

There is a further rounding problem which arises when calculations are done using variables. Numeric questions may take real numbers of up to nine significant figures and hence up to eight decimal places, but variables used in calculations round to four decimal places.

6.3.2. Handling of 'Don't know' and 'Refusal' answers

Blaise assigns the next value above the top of the range for a numeric question to 'refusal' and the value after that to 'don't know'. If the answer is displayed in text substitution in a different question or used to perform a calculation, the result is nonsense values. To avoid this it is necessary either to ensure that all subsequent questions and checks which make use of the value are routed on the question being equal to RESPONSE or to filter the answer via a variable. The latter may not be suitable for checks, because the facility of jumping to the incorrect question by highlighting it, is available only where the check is specified on questions rather than on variables. In either case the solution requires a great deal of extra programming. We think that Blaise ought to be able to recognise its own don't know and refusal codes and have some set way of dealing with them in displays and calculations.

6.4. The apparent unpredictability of the triggering of checks and calculations

We have come across instances where the triggering of checks and calculations appeared to us to be unpredictable. We are not suggesting that there are bugs or mistakes in Blaise, merely that it is rather difficult to fathom the workings of the program and that this is not well documented.

One problem with calculations relates to the fact that Blaise initialises variables each time it checks a questionnaire. Say you have a questionnaire where you want to keep a record throughout the session of the value that a question had at the start of the session. This could be used, for example, to check that the interviewer does not change the serial number. You might think that it would be possible to set up a variable (called, say, VSerNo) and initialise it at the start of the session with a statement along the lines of

```
IF VSerNo = 0 THEN
  COMPUTE VSerNo := SerNo
ENDIF;
```

But this does not work. Every time that SerNo is changed, VSerNo is also changed, because VSerNo is reset to 0 each time the questionnaire is checked. The solution to this problem is to use a PROTECT question and initialise it with the following statement:

```
CHECK
  IF (QSerNo = EMPTY) AND (SerNo <> EMPTY) THEN
     COMPUTE QSerNo := SerNo
  ENDIF;
```

Another problem appears to relate to checks tied to hidden questions. Sometimes checks that are hooked to hidden questions do not appear to be performed at the point where the hidden question appears in the ROUTE paragraph. We have not been able to isolate the precise conditions under which this occurs and we have often been put in the position of using trial and error, i.e. moving questions around within the ROUTE paragraph, until we hit upon a permutation that works. There is probably some very simple explanation to this, and we think this should be clearly explained in the documentation.

6.5. The lack of user defined keys

On the FRS, it is often the case that respondents do not have the information to hand at the time of the interview (e.g. their pay slip), but that they are willing to give the information to the interviewer later. The survey procedures allow for interviewers to recontact such respondents to collect these pieces of information. On a PAPI questionnaire, the interviewer would normally put a large star in the margin by such questions. We needed a way of emulating this in CAPI which would be available at all questions and easy to find afterwards, without cluttering up screens and answer lists. We decided to use the refusal key (]). By amending the CAPITEXT file, we have caused the refusal key to display a series of exclamation marks in the answer field. However, this means that we now do not have a universal refusal key.

It would be useful to have a special user defined key similar to [and] which could be used for such purposes. It would also be useful to have the ability automatically to review afterwards those questions that have been answered with this special key.

7. Conclusion

This paper describes some of the issues which have arisen in the implementation of a large and complex survey in Blaise. In general, we are very happy with the Blaise software and feel that it is suitable for such a survey. However, some ingenuity has been required to implement features such as complex data hierarchies, concurrent interviewing, repeated sets of questions, and edit checks. We hope that other Blaise users can learn from our experience. We have also outlined what we consider to be some of the limitations of Blaise, particularly in relation to data hierarchies, size constraints, numeric questions and variables, the apparent unpredictability of the triggering of checks and calculations, and the lack of user-defined keys. We hope that these comments will be taken on board in future versions of Blaise.

A POSSIBLE FUTURE DEVELOPMENT FOR THE BLAISE SYSTEM

Gary R. Dunnet
New Zealand Department of Statistics

1. Introduction

In this paper I will suggest that we could improve the editing of data by combining two recent advances in statistics. The first advance is the development of integrated survey processing computer systems, such as the Blaise System. The other advance is the development of simple graphical tools for data analysis, initiated by Tukey's [1] ideas of 'exploratory data analysis' (EDA), coupled with the development of computer technology to display graphics interactively.

Current editing systems, including Blaise, make only limited use of interactive graphics. I believe that interactive graphics will come to have an important place in the editing process. Many of my ideas here are based on the paper "Graphical Editing for Business and Economic Surveys" by Houston and Bruce [2].

My paper explores the importance of interactive graphics for editing data, and discusses how it could be integrated into a system such as Blaise. I will first outline the two different types of editing that are part of the editing process, "micro" and "macro" editing. After explaining the importance of graphics, I will explore the possible implementation of interactive graphics within "macro" editing. As support to my discussion I will outline the New Zealand interactive graphical approach. I will conclude with my thoughts on both where interactive graphical editing could fit into the Blaise System and how the survey cycle may be modified to accommodate this new approach towards "macro" editing.

2. The editing system

Statistical data collection is a complex process involving many stages, in which data on persons, households and businesses are collected by means of surveys and are then transformed into useful statistics. One stage is the process of data editing. The data editing process of any survey processing system may typically consist of :

1. Data entry and input editing,

2. Record Imputation, or Weighting Adjustments,

3. Output Editing & Estimation,

4. Analysis of results.

There is a clear distinction between the editing methods used in the various stages. At the first stage, "micro" editing is used where individual records are checked. The latter stages use "macro" editing involving a complete or near complete dataset. While individual records may be examined in the "macro" phase, the editing is done in the context of the rest of the data.

The philosophy behind editing at the data entry phase is simple : eliminating problems at the beginning of the data collection process reduces problems throughout the survey processing cycle. While any Editing System that follows this philosophy, including the Blaise System (Pierzchala [3]), does give better data quality at lower costs, it does not alleviate the need for editing at other stages. It has been Granquist's [4] and our experiences that "micro-editing may not always detect even serious errors", and that "over-editing" is likely to occur. Granquist [4] also found that "the macro-editing concept is a realistic alternative or complement to micro-editing methods, and can be applied during the processing of the data under the same conditions as computer-assisted micro-editing methods, which reduces the manual verifying work to a considerable extent". I do believe that "macro" editing may be used to complement "micro" editing, and I stress complement, by no means should an Editing System be solely based on "macro" editing. I believe that any enhancement to the Blaise Editing System, presently solely "micro" editing based, should be directed at the introduction of "macro" editing.

The approach taken to "macro" editing at the New Zealand Department of Statistics is to use interactive graphics. The reasons for the use of graphics is outlined in the following section, after which I will outline how Graphical Editing may be implemented inside "macro" editing. This will lead back to the earlier statement, that I believe that a Graphical Editing approach to "macro" editing may be used to complement the Blaise Editing System.

2.1. The Importance of Graphics

The role of "display" in data analysis is of extreme importance. Tukey [1] stated that "Graphics force us to note the unexpected; nothing could be more important".

Just as the availability of powerful mathematical tools has increased, so has the ability to perform arithmetic computations quickly and efficiently. Graphic techniques are useful to communicate the results of these arithmetic computations to the user. This leaves the user to think about what other things might be done and how results might be interpreted. With the increased power of graphics the process of interpretation is greatly enhanced.

Exploratory data-analysis is necessarily an iterative process, where many "tentative" analysis are performed, promising leads are followed up, some analysis discarded, and where strong, stable and believable patterns are desired. Although "pen and paper" may be used in this analysis it soon becomes apparent, as the number of arithmetic computations becomes sufficiently large, that extensive computerization of the effort is essential.

3. Graphical Editing within the Editing System

As outlined in section 2, the Editing System can be broken down into both "micro" and "macro" editing. This section will outline how Graphical Editing may be implemented inside "macro" editing, which is the approach taken at the New Zealand Department of Statistics. Although Graphical Editing is potentially useful at all stages of the survey processing cycle, I do not see any advantages in using graphics in place of the Blaise System during the "micro" editing stage of the survey processing system.

It should be noted that Graphical Editing can be classified into two modes of use : a "rigid" mode, controlled by menus and buttons, and a flexible "EDA" style, in which different types of plots can be created. The "rigid" mode uses a relatively small number of predetermined plots to uncover certain types of potential errors or problems. The New Zealand Department of Statistics approach is that of a "rigid" mode, but may (depending on the users requirements) incorporate "EDA" ideas at a later date by making it easy to bring data into suitable statistical packages. I will now give a detailed discussion on how Graphical Editing may fit into the three stages of "macro" editing outlined earlier.

3.1. Record imputation

After a sufficient quantity of data has been collected, any non-responses may be imputed using a statistical model. The statistical model may either be survey specific or from a pool of pre-defined imputation methods. Graphical methods would be used in the evaluation of these imputation methods. For example, graphical representations of the distribution of the imputed values may be produced. This sort of analysis would reveal whether imputed values have any unusual or unexpected properties. This type of graphical representation uses the "rigid" mode of editing.

3.2. Output editing

For Business surveys, output editing is the editing of data with the aim to validate the estimates (including the imputations), produced. To date the New Zealand Department of Statistics work in the area of output editing has been concerned with Business surveys, in the near future we hope to examine the area of output editing within Household surveys.

Output editing techniques serve as a backup to input editing techniques. With a reliable output editing system a more focused input editing effort can be achieved. By spending less time on errors which have little or no effect on the final estimates, more time is available to follow-up respondents who have a very large influence.

Occasionally, a sampling unit returns a value which is much larger than those expected at the time of sample selection, leading to an abnormally large weight. Such a sampling unit can have a large volatile effect on the final estimates. One method for handling the problem of large unexpected results is with the use of a robust estimator. Because the robust estimator adjustment may introduce bias, there is still a need for good output editing support. In the paper "Robust estimation and diagnostics for repeated sample surveys" Bruce [5] indicated both the power and relative ease of graphical editing at this stage. Bruce concluded that a fairly rigid graphical editing system along with robust estimators can handle problem estimation weights.

It may well be desirable to deal with "major" errors during the data collection phase, and leave "minor" errors to an automated batch editing procedure [6]. Although this may make a saving on labour costs, a wholesale-automated editing approach can introduce large biases into the estimates. Avoiding such bias would require ongoing validation, requiring the use of similar diagnostics to those outlined above.

Another aspect of output editing is that it can be used as a check on the editing system itself. For example, deficiencies in the input editing system can be analysed by examining the types of errors which go undetected. It may also be possible to identify when the input editing procedures are having no useful effect on the final estimates, however these types of problems may require a flexible interactive style.

3.3. Analysis of results

The routine analysis of results prior to publication is important. It is important to know whether unusual features are the result of genuine changes in the economy or are simply a reflection of a problem with the survey design or processing procedures. As outlined in section 2.1, routine availability of powerful graphical tools for exploratory data analysis may lead to the detection of previously unnoticed features.

4. The New Zealand approach

A new approach towards output editing based on interactive graphics and data analysis has been developed by Gary Houston at the New Zealand Department of Statistics. The thrust of our development has been towards the Output Editing stage within the "macro" editing system. This is where we saw the most gains could be made with the introduction of graphical editing techniques. The development of the system is based on the ideas of Bruce [5], Houston, and the survey sections themselves. The program is still under development but is being trialed by the business survey sections.

A general overview of the Editing System can be gained from the paper of Houston and Bruce [2]. A brief summary of the advantages of such a system is :

- The use of graphics allows a large amount of information to be displayed at once. "Multiple views" of the data can be simultaneously displayed.

- The use of a mouse allows the user to quickly move through the data identifying any points of concern.

- Buttons and menus can be used to control the user interaction, guiding the user towards consistent application of editing rules. At the same time, options can be provided to permit exploration and investigation.

- "Linked plots" showing the effects of the data value on the estimate can be graphically displayed simultaneously with the plot of the data.

- Historical data of panel surveys is used to show relationships across time as opposed to across variables.

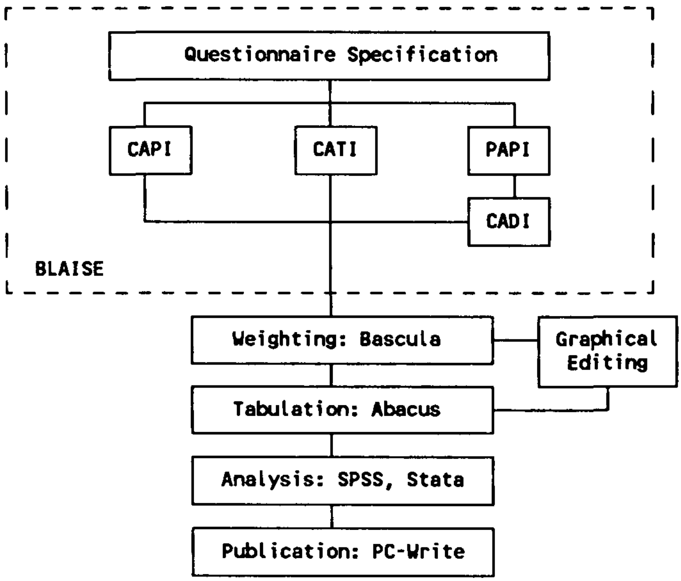
- The system is general purpose.

The current system runs on Sun workstations. The data extraction, manipulation and graphics are coded in C (originally Splus) and make use of the X11 window system. It should be noted that the programming involved is not trivial. The system can be applied to reasonably large datasets. It is hoped to construct tools for examining other potential problems, and to examine the usefulness of other types of plots.

5. Graphical editing within the Blaise system

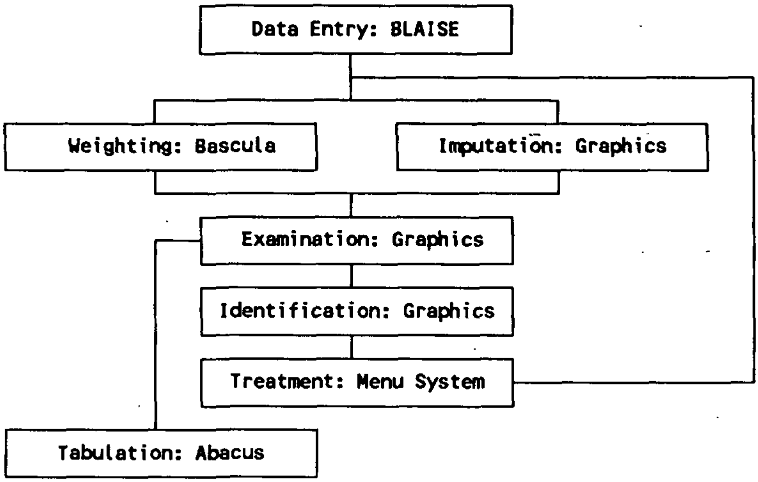
The logical place where Graphical Editing would fit into the current Blaise System is as part of or as an extension to the Estimation System i.e. Bascula. This is the idea in Figure 1. I have deliberately not included graphical editing with Bascula for three reasons. Firstly there may not be a Graphical Editing requirement within the survey process. For example, it is currently unknown whether graphical editing would be useful in a Household Survey. Secondly weighting may not necessarily use Bascula. Bascula allows for post-stratification to correct for non-response, whereas typically for business surveys imputation is used for non-respondents this may use graphics, as outlined in section 3.1. Thirdly I do not believe that the current Blaise System environment can "handle" the graphical requirements. To make full use of Graphical Editing the system may require, say, a 486 with a reasonably advanced operating system, which is solely dedicated to the function of Graphical Editing alternatively, as in New Zealand's approach a Sun workstation, along with Xterminals, linked into the Blaise System.

Figure 1. The Modified Integrated Survey Processing System



It may be more convenient to think of Bascula and Graphical Editing forming a loop with each other. This would lead to a typical Survey Processing Cycle similar to that outlined in Figure 2.

Figure 2. The Modified Survey Processing Cycle



6. Summary

The use of graphics for analysis of survey data shows promise for both survey monitoring and outlier detection. Graphics programs have the potential to develop into a primary tool for "macro" editing.

I believe that the combination of the Blaise System and graphical editing will produce a survey processing cycle with greatly increased data quality and productivity.

References

[1] Tukey, J. W. *Exploratory data analysis*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1977.

[2] Houston G. and Bruce A. G. *Graphical Editing for Business and Economic Surveys*. Draft paper, submitted to the Journal of Official Statistics 1992.

[3] Pierzchala M. *A Review of the State of the Art in Automated Data Editing and Imputation*. Journal of Official Statistics, Vol 6, No 4, 1990, pp 355-377, Statistics Canada.

[4] Granquist L. *A Review of some Macro-editing Methods for Rationalizing the Editing Process*. Proceedings of Statistics Canada Symposium 90, Measurement and Improvement of Data Quality, October 1990.

[5] Bruce A. G. *Robust estimation and diagnostics for repeated sample surveys*. Mathematical Statistics Working Paper 1991/1, New Zealand Department of Statistics, Wellington, New Zealand.

[6] Outrata, E. and Chinnappa, N. *General survey function design at Statistics Canada*. Bulletin of the ISI, proceedings of the 47th session, Vol 2, pp 219-238, 1989.

SOME APPLICATIONS OF WEIGHTING WITH BASCULA

Robert Göttgens
Netherlands Central Bureau of Statistics, Heerlen, The Netherlands

1. Introduction

More and more, statisticians use microcomputers for processing surveys. Since 1987 the Netherlands Central Bureau of Statistics (CBS) is developing the Blaise System for integrated survey processing, see e.g. Bethlehem (1991). Data collection, data editing, tabulation, and analysis can be carried out with this system on a microcomputer or on a network of microcomputers. In 1991 a new tool was added to this system: Bascula, a package for weighting sample survey data.

Survey data, obtained after data collection and data editing, are usually not ready yet for making inference about the population from which the sample has been drawn. The problem is that the data do not constitute a representative sample due to unequal selection probabilities and non-response. In order to correct for these effects, often adjustment weights are computed. Post-stratification is a well-known technique. Every record is assigned some weight, and these weights are computed in such a way that the weighted sample distribution of characteristics like sex, age, marital status, and area reflects the known distribution of these characteristics in the population. Two major problems can make application of post-stratification difficult: empty strata and lack of adequate population information. Research has been carried out at the CBS in order to improve weighting techniques. The result was a new general method for weighting, in which weights are obtained from a linear model which relates the target variables of a survey to auxiliary variables. Post-stratification is a special case of this method. Because of the generality of the method, different weighting schemes can be applied that take advantage of the available population information as much as possible, and at the same time avoid the above mentioned problems (Bethlehem and Keller, 1987).

Bascula is a general weighting package, running on microcomputers under the operating system MS-DOS. It combines several weighting techniques. In the first place, traditional post-stratification can be carried out. And if the number of empty strata is small, one can instruct the program to collapse (i.e. combine) these strata with neighbouring strata. In the case of many empty strata, or lack of sufficient population information, Bascula can carry out the linear weighting technique described above or apply multidimensional iterative proportional fitting (also called multiplicative weighting, or raking ratio estimation).

The user specifies the auxiliary information to be used for the weighting in the form of a model. This model will be confronted with the available sample and population information. If the specified model cannot be applied, Bascula will propose a simpler model. Bascula will carry out a complete post-stratification if possible. If not, the user has to decide either to carry out linear or multiplicative weighting. So if maximum auxiliary information is not available, or if it would result in unstable estimates, one has to choose between linear weighting and multiplicative weighting.

Three different weights are distinguished by Bascula: the inclusion weight, which can be attached to each separate record, the correction weight computed by Bascula and the final adjustment weight, the product of the former two. The resulting final adjustment weights can either be added to the data file, or be stored in a separate file.

It is possible to derive new variables or change the contents of existing variables by making use of the build-in recode facility. The options available in this facility are a subset of the options in a more general file manipulation package called Manipula (Hofman et al., 1990). Manipula can be used if the recoding facility in Bascula is not sufficient.

2. Some applications of weighting with Bascula on CBS-statistics

2.1. The National Voters Survey (NVS)

Before and after the parliamentary elections of 6 September 1989, a survey took place into voting behaviour and backgrounds. This National Voters Survey combines background items with questions scaling opinions on a variety of party-political subjects. To analyse characteristics of the Dutch electorate, which influence party choice, a structural equation model was developed (Schmeets 1991).

The NVS 1989 consisted of two rounds: one interview before and one interview after the election. Eventually 1506 persons were interviewed face-to-face with the aid of a Blaise CAPI program, which had as major advantage the possibility of cleaning the data during the fieldwork. The response being low, a weighting procedure was needed to correct the response pattern for the distribution of voting behaviour, age, sex, marital status, degree of urbanisation and region in order to enhance as much as possible the statistics on voting behaviour and attidudes. So the following characteristics were used to make different weighting models: voting behaviour (8 categories), age (7 categories), sex (2 categories), marital status (4 categories), degree of urbanisation (4 categories) and region (4 categories). By means of these auxiliary information the following weighting models were constructed and evaluated (Schmeets and Molin, 1990):

1. Sex + Age + MariStat + DegOfUrb + Region
2. Sex x MariStat + Sex x Age + DegOfUrb + Region
3. Sex x MariStat + Sex x Age + DegOfUrb x Region
4. VotBehav + Age + Sex + MariStat + DegOfUrb + Region
5. VotBehav
6. VotBehav + Age

The notation used in these models has some analogy with analysis of variance. An 'x' means that variables are crossed, i.e. the population distribution in the table obtained by crossing the variables is used for the computation of weights. A '+' means that only marginal distributions are used. For example, the model 'Sex x MariStat + Sex x Age' means that the two population tables of Sex by Marital status ans Sex by Age are used, but not the three-way table Sex by Marital status by Age.

For all these weighting models the weights were calculated using the additive weighting method. The different results of the models were compared and after extensive analysis and discussion one decided finally to use the fourth model. It was very evident that one of the major advantages of Bascula during the weighting of this sample turned out to be the flexibility and fastness of selecting and calculating the different weighting models. In earlier years of the NVS one had to specify all desired weighting models in writing and offer this to the statistical methods department. So tailor-made software had to be made for each weighting model. Bascula needs only a few minutes to calculate a weighting model as described above (n = 1506, on a 386 machine), so it saves both time and much programming efforts.

2.2. The Family Expenditure Survey (FES)

The CBS has conducted an annual Family Expenditure Survey ever since 1978, representative for all households in the Netherlands. In the FES 1991 eventually 2859 households were interviewed face-to-face with the laptop computers using a Blaise CAPI program and paper and pencil questionnaires. Like in the NVS described above, to attain a good representative sample of the population one needs to apply some weighting method. In this survey the following characteristics were used as auxiliary information: the income of the household, the number of persons in the household, the socio-economic category and sex of the head of the household and the share of households living in owner-occupied dwellings. By means of these characteristics some 'new' auxiliaries were defined: a combination of income and the size of the household IncSize with 13 categories and a combination of income of the household and socio-economic category of the head of the household IncSec with 37 categories. With this auxiliary variables the following weighting scheme was used:

HireOwn + IncSize + Sex + IncSec

In this case, for two reasons the chosen weighting method concerned multiplicative weighting: first the final adjustment weights should always be positive, which is guaranteed in the multiplicative weighting method. The second reason concerned the possibility of defining lower and upper weight limits in the multiplicative weighting module: by doing so it is guaranteed that the final adjustment weights lie within the range defined by the lower and upper weights.

References

Bethlehem, J.G. and W.J. Keller, 1987, Linear Weighting of Sample Survey Data. Journal of Official Statistics 3, pp. 141-154.

Bethlehem, J.G., A.J. Hundepool, M.H. Schuerhoff and L.F.M. Vermeulen, 1989, Blaise Blaise 2.0 / Language Reference Manual. CBS report, Netherlands Central Bureau of Statistics, Voorburg.

Bethlehem, J.G. and W.J. Keller, 1991, The Blaise System for Integrated Survey Processing. Survey Methodology Journal 17, pp. 43-56.

Göttgens, Robert, Bert Vellen, Marly Odekerken and Lon Hofman, 1991, Bascula, version 1.0 / A weighting package under MS-DOS. User Manual. Netherlands Central Bureau of Statistics, Voorburg, The Netherlands.

Hofman, L.P.M.B., H.W.J.M. Vellen and J.H.A.M. Meertens, 1990, Manipula, Version 1.0 / File manipulation under MS-DOS, User Manual. CBS report, Netherlands Central Bureau of Statistics.

Schmeets, J.J.G., 1991, Party choice in the Netherlands: Model, weighting and political interest, Proceedings of the twelfth meeting of the 'Working group structural equation modelling' at the Netherlands Central Bureau of Statistics in Heerlen, 4 and 5 april 1991.

Schmeets, J.J.G. and E.J.E. Molin (1990): Constructie en evaluatie van weegmodellen in kiezersonderzoek, CBS report, Netherlands Central Bureau of Statistics, Heerlen.

BLAISE INTEGRATION IN IBGE'S DATA PROCESSING ARCHITECTURE

Reina Marta Hanono
IBGE/DETEC/DISIA, Rio de Janeiro, Brasil

1. IBGE

The INSTITUTO BRASILEIRO de GEOGRAFIA e ESTATISTICA (IBGE) is the responsible institute for all kinds of censuses (Economic and Demographic) and several continuous (monthly and annual) surveys. The responsibility of processing and disseminating the available data to the society involves much time and sometimes a great amount of money.

2. Data processing strategy

The developments in computer hardware and software have given rise to new possibilities for organizing automation in the processing of survey data in statistical bureaus.

Following the new trends in automation, at IBGE we began to study our data processing strategy, guided by the following aspects: decentralization, portability, independence and integration of both hardware and software, in order to bring the end-users next to the process of their survey.

3. Generic scheme

As a result the department of technology proposed a 'Generic Scheme' [1] which takes care of the process of planning and production for the statistical survey, composed of two systems and six functions to perform the major steps in statistical survey processing.

The two systems are:
- the 'Data Base Server System' (DBSS)
- the 'Specification and Project System' (SPS)

The DBSS takes care of the maintenance and the retrieval of data and meta data information.
The SPS is responsible for the project phase and standardization of the interfaces of the several tools selected to perform the data flow functions.

The six functions related are:

- data collection and capture
- editing and correcting
- estimation and expansion
- sampling
- analysis
- product extraction, which will be able to take care of data tabulation and publication

To each function is associated a proper set of standardized tools, which have the responsibility to treat the statistical data using the DBSS, taking the necessary parameters from the SPS.

4. Blaise integration in the Generic Scheme

Looking at the editing and correcting function, we developed software called CRIPTA [3], which generates portable applications for centralized and decentralized environments (UNIX and DOS). Still, we are really interested in integrating Blaise in our Generic Scheme, because of its facility and user friendly interface, for generating CAPI, CATI and CADI applications that could be used for processing the monthly or annual surveys.

The integration would be done by incorporating the meta information generated by Blaise, that of the quest and check paragraphs of the Blaise questionnaire, in the enterprise meta data dictionary. This data dictionary contains all the survey's meta information, such as data layout and editing (check) rules, which is part of the DBSS. In that way we would like to generate the application dictionary, which contains the logical layout of the record to be processed, directly from the meta information generated by the quest paragraph and vice versa. This will allow us to use Blaise's facilities for interacting coding and tabulation (Abacus) as tools of the Generic Scheme in any survey pre-process by Blaise or not.

Moreover we would like to generate the group of editing rules from the information generated by the Blaise check paragraph. As in the Blaise check paragraph we admit direct or consistency editing rules, which are defined in a standardized form, 'Automated editing plan' (Plano de Critica Automatizado), called PCAUT [3] in the DBSS. This permits the automatic generation of CRIPTA [4] editing and correcting applications and of DIA [5] imputation applications (this interface has been developed at IBGE in order to integrate DIA in the Generic Scheme).

The PCAUT form is defined by specific items in which the user (survey specialist) must specify the editing rules. They are visualized as a concatenation of:

- a reference identification,
  which identifies the rule and the error message
- a 'process',
  where the user must specify in the CRIPTA language the process to be carried out and the condition to be checked so as to verify the rule
- an 'effect',
  the action to be performed by the system in response to the condition to be checked, and optionally the list of variables to be printed in case of error

We present some examples of editing rules written in PCAUT form and using CRIPTA's language, where the rules specified are written in the negative form, that is if the condition is true (OPC = T) the record is on error, but that can also be written in any other way:

[REFERENCE]	[PROCESS]	[OPC]	[EFFECT]
E10	' Invalid interview type ' IF INVALID(V106);	T	V106; ERROR;
E20	' Invalid household type ' IF V201 ^= All(1,2,3,5);	T	V201; ERROR;
E50	' Total of persons less than number of person older than 9 ' IF V107 < V108 ;	T	V107,V108; ERROR;
E60	' Total of persons older than nine different from number of persons older than 9 ' IF V108 ^= ' ' & V108 ^= FREQUENCY(V100 =3 & V805 > 9);	T	V108; ERROR;

The integration we talk about will enable us to use CADI, CAPI or CATI applications integrated with other processing tools such as CRIPTA [4], for manual or automatic correcting applications, DIA [5] for automatic correcting applications, using Fellegi & Holt methodology, and any other tool incorporated in the architecture of the GENERIC SCHEME. Today at IBGE, to specify an application using CRIPTA or DIA systems, the user has to specify the application dictionary and the group of editing rules in only one way in the DBSS.

Moreover the integration will allow us to use Abacus and/or any other Blaise tool for the processing of statistical surveys. We are really interested in Abacus because it is a very powerful tool for the tabulation process and we want to use it for producing or testing tables in any survey, pre-processed by Blaise or not, importing Blaise's meta information from the enterprise data dictionary. It will also permit us to maintain the consistency of the survey documentation in the DBSS.

It seems to us that it is a little hard to give a detailed description of the integration we propose in this paper. We hope that this brief description will be sufficient so as to understand our intention.

Clearly the integration will have three stages:

- integration of the questionnaire description (Blaise quest paragraph) into the DBSS survey dictionary;
- integration of the check rules (Blaise check paragraph) into the DBSS survey editing rules;
- integration of the route paragraph (Blaise route paragraph) into the DBSS. At the moment we don't have an entity for them in the DBSS meta data base, but we are sure that it would not be difficult to define it, and that it will be very useful for integrating different tools for data capture.

Bibliography

[1] Guedes A.P. and Cabral M.S.S., 1992, A GENERIC SCHEMA for survey processing at IBGE, (paper accepted by COMPSTAT 1992) available at IBGE/DI/DETEC.

[2] Blaise user's manual, Netherlands Central Bureau of Statistics, Voorburg, The Netherlands.

[3] CRIPTA user's manual, IBGE/DI/DETEC, Rio de Janeiro, Brasil.

[4] Hanono R.M. and Barbosa D.M.R., 1992, A generic system for editing and correcting applications (paper accepted by SGCSA (1992) available at IBGE/DI/DETEC.

[5] DIA user's manual, INE, España.

USAGE OF MANIPULA AT THE NETHERLANS CENTRAL BUREAU OF STATISTICS

L.P.M.B. Hofman
Netherlands Central Bureau of Statistics, Heerlen

1. Introduction

In October 1989 the CBS started the development of the general file manipulation program Manipula. One of the reasons for developing this program was the shift of the statistical production process from the mainframe computer to the local area network (LAN). On the LAN no general file manipulation program was available. At some departments a new Pascal program was written for each file manipulation. Some departments tried to develop their own manipulation programs. Everybody wrote their own sorting program. This was of course not very efficient and it did not contribute to the CBS standardization. For efficient implementation of the statistical production process on the LAN at the CBS a general program was a necessity. One of the goals of such a program should be to relieve the user of all the extra around the manipulations. The user must be able to concentrate on his manipulations without having to worry for instance about opening and closing files.

In February 1990 a test version was installed on the LAN. Based on the comments and wishes of the users a lot of changes and extensions were made and in November 1990 version 1.0 was released. One of the important features of this version was the possibility of reading data stored in Blaise files directly. In 1991 a version was released for the mainframe computer. This version has roughly the same possibilities as the LAN version.

The usage of the program at the CBS is widespread. Because the program is so versatile, a lot of different (and for the makers of the program) unexpected applications can be found. Manipula is also used in situations for which it was not primarily intended, for instance tabulation and checking. There is some overlap between Manipula and other tools in the Blaise system and there even is some overlap between Manipula and Blaise. In practice Manipula has proved to be a jack-of-all-trades. By combining different kind of usages of Manipula and other standard tools in the Blaise system complex statistical production systems can be and have been built.

It would have been possible to describe in this paper one of the statistical production systems which makes use of Manipula, but that is really a task for that system's author. In this paper a different approach is chosen. Some examples are given of the kind of usage of Manipula encountered by the developers of Manipula during consulting and support at the CBS. The following kinds of usage will be described briefly: preparation for Abacus, conversion from ASCII to Blaise, conversion from Blaise to ASCII, reports, tabulation, linking data, manipulation of several input files in one run and checking data.

2. Usage of Manipula

2.1. Preparation for Abacus

Sometimes a table has to be produced from data not directly suited for tabulation. Abacus however offers only a limited number of data manipulation functions, for instance only one-dimensional recodings are possible. Manipula has proved to be very handy and powerful if the manipulations are too complex for Abacus. Manipula can prepare the file to be tabulated with Abacus.

2.2. Conversion from ASCII to Blaise

The CBS receives a lot of data in machine-readable form (on tape, diskette, by telephone). Blaise is often used for further editing. But how to get the ASCII data in the right conversion format? Manipula can do this job fairly easily. With the Setup generator a Manipula description of the ASCII import file can be generated. This description is used in the output paragraph in the Manipula setup. A description of the received file is always available (unless something is very wrong). This description is translated into a Manipula file description and used in the input paragraph. The manipulate paragraph offers possibilities to compute new variables, make a selection and so on. It is also possible to combine several input records into one output record or to create several output records from one input record. This can be important in case subfiles are used. If there is no unique key available in the input record it can be computed, for instance by using the record number.

2.3. Conversion from Blaise to ASCII

Manipula makes it possible to access data stored in a Blaise file directly. It is the only tool in the Blaise system that gives the user full control over the Blaise data. Because Manipula does not use the meta-data of a questionnaire the user has to take care of a number of aspects normally taken care of by the Blaise conversion program Convert. An example of this is the translation of the 'don't know' and 'refusal' codes. If the questionnaire is small (which is often the case) or if the codes are not accepted by the data entry program no special provisions are necessary. In spite of the possibly negative sides of Manipula as a tailored conversion tool it is used a lot for this purpose. An example is the Labour Force Survey (a huge questionnaire!). During the conversion process of this survey all open answers are replaced by a code which indicates whether an answer has been given.

2.4. Reports

It is very easy to display the contents of records or parts of records in a printable ASCII file with Manipula by using a print paragraph. In the Manipula setup headers and footers can be defined which are displayed on a page together with the contents of a number of (selected) records. The headers and footers need not be fixed text but can contain variables. In this way the user has full control over the page contents.

Examples of this usage can be found at various places. For instance, Manipula is used to generate reports with visit instructions for different field employees in the administrative application described in Lammers (1992).

2.5. Tabulation

In spite of the presence of Abacus, Manipula turned out to be used for tabulation purposes also. Sometimes the layout of a table is prescribed by an external client. An example of such a client is EUROSTAT, the statistical agency of the EC. For some surveys EUROSTAT provides a detailed descriptions of the layout of the tables. With Abacus these tables cannot always be made. Using sorting and summing, plus a print paragraph the job usually can be done with Manipula. Although it takes longer to produce a table with Manipula then with Abacus there is no limit to the size of the table that can be produced. Examples of such tables are the Energy Tables of the Foreign Trade Statistic. Descriptions of the different codes used in the tables are usually obtained by linking data files.

2.6. Linking data

Manipula offers an easy way of linking data files of different sources and of different types. File types supported for linking are Blaise, ASCII and Index (created with the external tool of the Blaise system). It is very easy to instruct Manipula to look up data in such a link file. The link data files can be used for different purposes: recoding data, joining files, for making a selection and so on. Linking data files is also used for partial update of a Blaise form. In a Manipula setup an ASCII file with update values of some fields in a Blaise form is linked with the Blaise data file. The output file contains the completely converted Blaise form with the updated values. This form is imported in Blaise by using the conversion tool.

2.7. Manipulation of several input files in one run

By using a wildcard in the input filename Manipula offers the possibility of carrying out the manipulations on all data files which conform to the wildcard in one run. An example is the situation where data is stored in several files, for instance for each week a different file, and a report has to be made concerning all files. The setup which can be used for one file can also be used for all files simultaneously.

2.8. Checking

Manipula offers the possibility of carrying out various checks on a data file. Checks can be made on the level of a data field (is it numeric?) but they can also be relational checks within one data record. It is also possible to carry out checks accross several input data records. We have a number of examples where Manipula is used as a checking tool. In many cases it would also have been possible to do the checking with an integral check in Blaise CADI.

Manipula is used for checking ASCII files. If the checks are simple this is sometimes more efficient then using Blaise because data does not need to be converted to the Blaise format. But Manipula is also used to do extra checks on data stored in Blaise files. In this case a report is generated with the keys of the cases that need to be examined interactively. There is also an example of a complex check program implemented in Blaise on the LAN. The same checks are carried out on data stored on the main frame using Manipula.

References

Lammers, J.P.M., 1992, The use of Blaise for administrative purposes. Proceedings of the First International Blaise Users Meeting, 14-16 October, Netherlands Central Bureau of Statistics, Voorburg.

BLAISE AND THE AGRICULTURAL CENSUS
Batch processing on a CDC mini computer

Hans de Jong
Netherlands Central Bureau of Statistics, Voorburg

1. Abstract

When a CBS data processing system, like the one for the Agricultural Census, has to be built or rebuilt, the use of software packages like Blaise is always considered and encouraged. In this case the substantial size of the data files presents some severe difficulties if all error checking has to be done on personal computers. Nevertheless it turned out that a batch version of Blaise, running on a minicomputer network, could be applied successfully in the data validation process, starting with the 1992 census. The same Blaise questionnaire was used both for the integral check on the CDC minicomputer network and for the interactive data editing on PC's in a LAN. This paper gives an overview of this Blaise application and discusses some related problems and solutions.

2. The Agricultural Census

This yearly Census, organised by the Ministry of Agriculture, Nature Management and Fishery and the Netherlands Central Bureau of Statistics examines about 125,000 agricultural holdings. The questionnaire is made up of approximately 300 questions about labour force, areas of crop, and size of livestock. The census results in the publication of a number of basic figures, which are used for crop estimation, and several more detailed statistics, giving insight in the size and structure of the agricultural holdings. The edited data files are used for other surveys and further analysis.

There is a considerable demand for early provisional figures, within a few weeks after the questioning period. To be able to respond to this demand it is necessary to enter and edit the huge bulk of data in a very short time.

3. Never trust external data

From 1992 on the data entry process is no longer executed by the CBS itself, but accomplished by the Ministry of Agriculture, using Oracle. Both institutions agreed upon the specification of the questionnaire and the error checking to be used in the data entry process. The agricultural holders are invited to fill in the questionnaires at home and present them at local meetings, where they are entered into the Oracle system. In this way it is possible to detect and correct errors immediately. Further investigation by the Ministry can lead to changing some of the data. The edited data is offered to the CBS.

Even though the data is checked by the Ministry, the CBS wants to check it again, on the grounds that the Ministry cannot guarantee the data to be valid, because:

- the data is kept and sent to the CBS from provincial computer sites;
- it is not improbable that the validating software used by the Ministry is incomplete or even bugged;
- most of the checks by the Ministry are just warnings, and they might be suppressed at large by some of the interviewers;

To get some idea of the quality of the data the CBS also wants to do some additonal checks; if the quality of the data is high, the checking by the CBS will not yield a lot of errors, and so not much edit work is generated. If the quality is less than expected, the errors can be corrected or, which is more likely, the Ministry can be asked to adjust its error checking programs or procedures.

4. Synchronising two data editing processes

The data entry process by the Ministry takes place mainly from April until the middle of June, but data can be entered and modified till the end of August. If the CBS waited for the Ministry data to be complete before starting the extra data editing, the provisional figures would come out much too late, as they are required in July. Starting earlier, however, raises some problems too: the data of a particular holding could be modified by both the CBS and the Ministry at the same time. Because the CBS is only able to bring about some statistical corrections, it is very interested in receiving the corrected data from the Ministry. One could ask the Ministry to pass on just modifications instead of all available data, but the Ministerial data systems are not capable of doing so, and earlier experiences with external registrations have proved that this approach does not work. A better solution is to ask the twelve provincial sites to forward to the CBS all available data three or four times during the questioning period. By comparing the successive data files the CBS can determine if the data of a particular holding have changed. New and modified holdings can now be selected and joined with the already existing and partially corrected data files. All existing data of a holding will be substituted by the modified data, thereby loosing some of the CBS corrections, and necessitating the checking and correcting of the same errors again.

The Ministry told us it was not possible for the data of a holding to be deleted. This, however, often does happen. A temporary solution for this problem has been found.

5. To Blaise Or Not To Blaise

After some prototyping it turned out that, due to the size and complexity of the questionnaire and the validation requirements, importing and integral checking of the data in Blaise could not be done on a PC at a rate above 25 forms per minute. This would lead to a typical computing time of 6 to 7 hours for the data of one province, which is valued as too long (what time would be acceptable?). Another problem would be the necessary disk space of 15 to 30 Megabytes per province. A solution was requested in which the integral check could be run on the minicomputer network, and only the faulty data would have to be edited on a PC.

A possible way out of this problem, but not a favourable one, would have been to build a program for the integral error checking using a third generation language like Cobol or Pascal, and using Blaise for the data editing. This would mean to write two identical error checking programs in two languages. Twice as much work, and in later years twice as much maintenance effort. Furthermore, it would be likely that these programs would differ in small details, giving rise to unexpected problems.

A solution where all of the processing, including the interactive data editing process, would take place in the minicomputer environment was out of the question because it would seize too much of the available resources. The same was true for using Oracle in client/server mode, or trying new technology like a LAN batch server.

As it was known that the Blaise team was developing a batch version of Blaise for running the integral checking process on the CDC minicomputer network, this option, together with data editing in Blaise PC, was seen as the best alternative. Use of this product was agreed upon, provided that it would be operational at the appointed time and perform well enough, compared to the PC version. Passing the deadline would imply that a Cobol program had to be written after all. Care was taken that the other data processing systems, to be built for the Agricultural Census, were able to co-operate with both alternatives.

The Blaise team succeeded in producing a usable and well performing version just in time, and was able to eliminate a few bugs in short time, so fortunately there was no need to do the extra work of building a Cobol program.

6. Developing a Blaise Batch application

To be able to cope with the many possible file formats on a minicomputer, a Blaise Batch program is embedded in a Cobol program, generated by Manipula. The Manipula setup can be adapted by the systems developer. Inside Blaise, just a plain file format is used. For the parsing and compiling of the questionnaire some special actions have to be undertaken. Constructing the batch program is done in the following way: first a regular Blaise specification for a PC CADI-machine is developed and tested. After clearing all the small errors, the developer calls the Blaise parser with a special parameter, telling it to generate a number of C source files and header files. These files are transported to the minicomputer network, where they are compiled and stored in an object library. Manipula is used to generate a Cobol program with which it is possible to call the C subprogram. All this is completed with a tailor-made JCL file. Cobol is used because it has - on CDC computers - an extremely good file input and output performance.

The Blaise team has the intention to include the batch option in the Blaise menu system and to have these actions be executed automatically.

7. The performance of Blaise Batch

The performance of Blaise Batch, compared to Blaise PC, was an important factor in the decision to use the package. The performance was expected to be better, but how much better it would be was not known. It can be fascinating to compare the performance of a package like Blaise on different hardware configurations. Yet we have to be careful drawing universal conclusions, because there are many interfering circumstances, some of which are CBS-specific. The brand and type of PC, operating this PC without a local harddisk in a LAN instead of using it as a stand-alone, the performance of the LAN and the fileserver, and the number of other LAN users will have an influence on the performance of Blaise on PC, while the sharing of a minicomputer with other users and other batch jobs will have a major effect on the performance of Blaise Batch.

Moreover, the two programs do not carry out exactly the same job. In this application the PC version converts a main file and subfile from ASCII to Blaise and performs the integral check. The mini version has to convert a main file with a variable length to a fixed-length format, do the checking, and write the dirty and suspect forms to a main file and subfile in ASCII.

After a number of production runs it turned out that the Olivetti M300 can process about 24 forms per minute, while the minicomputer network reaches an average rate of 103 forms per minute, ranging from a minimum of 36 to a maximum of 343 forms per minute. Of course, processing a different questionnaire will give different, and in many cases better, results on both configurations.

8. Restrictions of Blaise

A frequency table of the errors and warnings was one of the requirements for the validation process. As Blaise does not produce this kind of control information it was necessary to create a special error block in the Blaise questionnaire and do some extra computing.

Unfortunately Blaise Batch does not produce the file format needed by Blaise PC, so instead of just transporting the Blaise files, one now also has to convert from Blaise Batch to ASCII and from ASCII to Blaise PC, and to repeat the integral checking. This can be a nuisance if the number of faulty records is relatively large.

9. Conclusions

It is possible to apply Blaise for validating large data files. Using Blaise Batch on a minicomputer network for the integral check and selection of dirty and suspect forms, and Blaise PC for the interactive data editing is favourable, compared to using the PC version only, because:

- the run time of the integral checking process is shorter;
- the integral checking can be done at any time, especially at night;
- several checking batch jobs can be started at the same time;
- the PC can be used for other activities;
- disk space on the PC or LAN can be preserved;

As the same Blaise dictionnaire is used in Blaise PC and Blaise Batch, development and maintenance time can be saved. At this time a tailor-made system still has to be constructed around the Blaise Batch application.

Of course there are a few drawbacks:

- in this application it is not easy to edit a clean form, as this never reaches Blaise PC;
- managing an application on two different hardware configurations, and controlling the communications between these configurations can form an extra burden for the users.
- the designer has to be acquainted with the internals of Blaise to be able to write the main program in Manipula.

Nevertheless it should be obvious that Blaise Batch will be used in many new or renewed statistical data processing systems at the Netherlands Central Bureau of Statistics.

ELECTRONIC DATA PROCESSING IN OFFICIAL STATISTICS
Recent developments and future prospects

Wouter J. Keller
Netherlands Central Bureau of Statistics, Voorburg

1. Introduction

Recent developments in Information Technology (IT) have a large impact on the way in which national statistical offices process their data. In particular, the introduction of microcomputers has caused changes in the organization and the activities of the Electronic Data Processing (EDP) department and the subject matter departments. Simple and straightforward survey data processing can now be concentrated in the subject-matter departments, whereas design of complex information systems or of generalized software packages remains the task of the specialist of the EDP department.

The effects of a more intensive and sophisticated use of IT are twofold: On the one hand, there is an increase efficiency and data quality, but on the other hand, they are also new problems. Due to the decentralisation of data processing activities there is a need for data sharing and standardisation of tools. Furthermore, since most data processing can now be carried out by the statisticians in subject-matter departments, there is also a need for an integrated set of user-friendly data processing tools. All this asks for a re-orientation of the role of the EDP department. But it also asks for changes in the organization of official statistics.

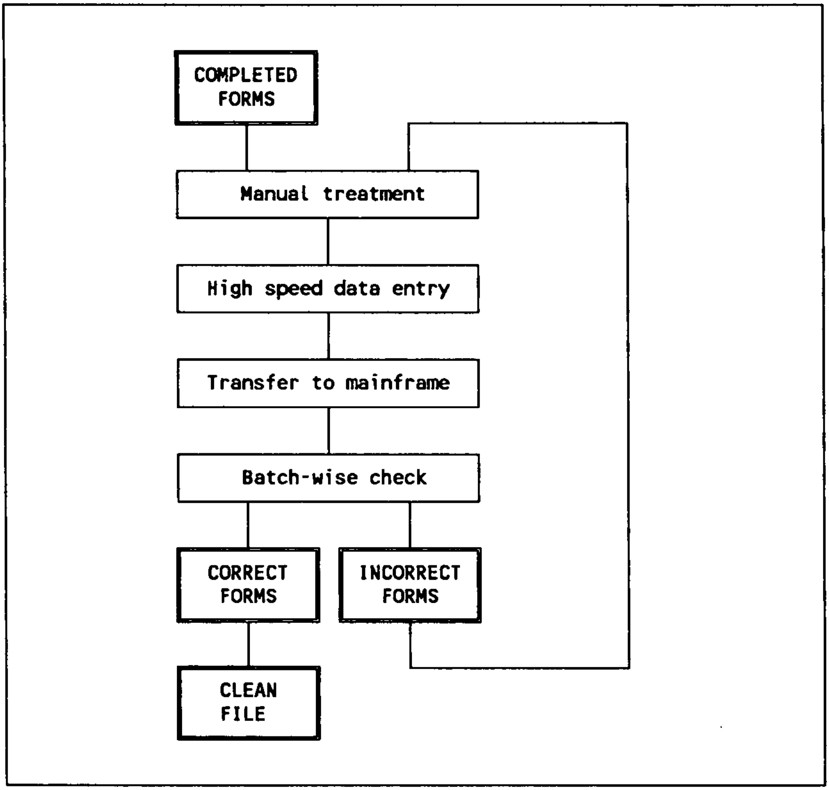
This paper describes some of the opportunities offered by the new IT developments. It discusses the consequences of changes for both the statistician as well as for the EDP specialist. In particular, it concentrates on the use and impact of several new ways of survey processing such as Computer Assisted Interviewing. The paper is a slightly adapted version of a paper presented at the "Third Independent Conference of the AOS", which was held in Ankara, on 22-25 September 1992.

2. Survey processing the traditional way

Traditionally, survey processing is organised around one, big, mainframe computer, with a central EDP department, a central processing unit, central data, central EDP management, and above all, central system development. The organization of the statistical production process reflected this central approach: data collection, data entry, data editing, imputation, tabulation and analysis were separate processes, often carried out in a batch-wise manner, with several batch cycles in each process. These separate processes were often handled by separate groups or departments, on separate computer systems, and with different types of specialisation. As an example, let us look at one important step in the processing of survey data: data editing.

Although the data editing processes differ from survey to survey, still some general characteristics can be observed which hold for nearly all surveys. The traditional editing process is summarized in figure 1. After collection of the forms, subject-matter specialists checked the forms for completeness. If necessary and possible, skipped questions were answered and obvious errors were corrected on the forms. Sometimes, the forms were manually copied to a new form to allow for the subsequent step of fast data entry. Next, the forms were transferred to the data entry department. Data typists entered the data in a dedicated data entry computer at high speed without much error checking. After data entry, the files were transferred to the mainframe computer system. On the mainframe an error detection program was run. Except for errors that were automatically corrected, detected errors were printed on a list. The lists with errors were sent to the subject-matter department. Specialists investigated the error messages, consulted corresponding forms, and corrected errors on the lists. Lists with corrections were sent to the data entry department, and data typists entered the corrections in the data entry computer. The file with corrections was transferred to the main-

Figure 1. The traditional approach to data editing



frame computer. Corrected records and already present correct records were merged. The cycle of batch error detection and manual correction was repeated until the number of detected errors was considered to be sufficiently small. After the final step of the editing process the result was a 'clean' data set, which could be used for tabulation and analysis. Detailed investigation of this process lead to a number of conclusions. These conclusions are summarized below.

In the first place, various people from different departments were involved. Many people dealt with the information: respondents filled in forms, subject-matter specialists checked forms and corrected errors, data typists entered data in the computer, and programmers from the computer department constructed editing programs. Transfer of material from one person or department to another could be a source of error, misunderstanding and delay.

In the second place, different computer systems were often involved. Most data entry was carried out on minicomputer systems, and data editing programs often ran on mainframes. Transfer of files from one system to another caused delay, and incorrect specification and documentation could produce errors.

In the third place, not all activities were aimed at quality improvement. A lot of time was spent just on preparing forms for data entry, and not on correcting errors. Subject-matter specialists had to clean up forms to avoid problems during data entry. The most striking example was assignment of a code for 'unknown' to unanswered questions.

Another characteristic of the process is that it was going through "macro cycles". The whole batch of data was going through cycles: from one department to another, and from computer system to another. The cycle of data entry, automatic checking and manual correction was in many cases repeated three times or more. Due to these macro cycles, data processing was very time consuming.

Finally, the structure of the data had to be specified in nearly every step of the data editing process. Although essentially the same, the 'language' of specification could be completely different for every department or computer system involved. The questionnaire itself was the first specification. The next one was with respect to data entry. Then, automatic checking program required another specification of the data. For tabulation and analysis again another specification was needed. All specifications come down to a description of variables, valid answers, routing and possibly valid relations.

3. The impact of information technology

New ways of processing surveys have become possible during the last decade. The main driving force behind this development is the revolutionary change in the field of information. If we look at the three activities data processing, data storage and data transport, the changes in the first one, data processing, are the most appealing. These changes are mainly the result of the introduction of the single chip computer in the late seventies, replacing mini and mainframe computers. Using the performance of the old VAX780 minicomputer as the standard (1 MIPS), the present processor chips run 20 times (Intel X86) to 50 times (RISC) faster. During the last years, the increase in performance per dollar was approximately 60% a year. At the end of 1992, 100 MIPS chips are expected!

It is not surprising that many organizations try to take advantage of the dramatic improvements offered by IT. IT enables us to work more in teams than in departments. This contrasts the more traditional oriented Taylor-like organization, which relies on specialisation, batch-work and sequential processing. The team approach today, as seen, e.g. in the automotive and aerospace industry, is a more integrated, interactive and parallel approach where several of the traditional stages are done nearly simultaneously by the same group of people ("concurrent engineering"). This is made possible because more powerful computers allow for more interactive work, while at the same time computer networks allow for the sharing of data among team members.

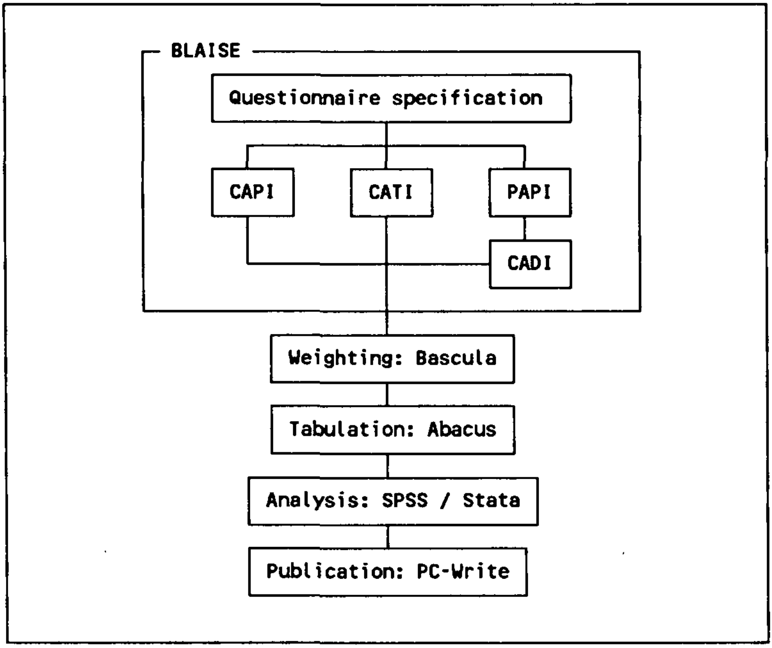
In the past, mainframes and large, serial, batch cycles were often the only means to process large surveys. Hence the mainframe was in the middle and everything was organised around it. However, in a modern organization there are no computers (mainframes) surrounded by people, but people surrounded by (micro-)computers, so the human individual instead of the computer has become the focus point.

The changes in IT also had an enormous impact at the organization of official statistics. Nowadays, survey processing can be done interactively instead of batchwise, allowing for much smaller development cycles and integration of several processes into one. Often processing is done on a record-oriented instead of batch-oriented. Complete decentralized processing, even as part of interviewing in the field, is made possible by this change in technology.

Decentralized, record-oriented and interactive processing is not only typical for the new approach to data collection and data editing, but also for other activities in survey processing such as coding, imputation, weighting, and tabulation. Error checking and correction has become an intelligent and interactive process, to be carried out by a subject-matter specialist on her microcomputer. The traditional batch process in which the data set is processed as a whole, is replaced by a record-oriented process in which records are dealt with one at the time. So, macro cycles are replaced by micro cycles: the subject-matter specialist enters a record, and keeps working on it until no more error messages are produced. When she is ready with a record, it is should be error-free and ready for further tabulation, analysis, etc.

Preferably, all software required for survey data processing is part of an integrated system. An integrated system for survey processing should be based on a powerful language for the specification of questionnaires. This specification is the 'knowledge base', containing all knowledge about the questionnaire and the data. The system should be able to exploit this knowledge, i.e. it must be able to automatically generate all required data processing applications. On the one hand it means the automatic generation of software for data collection, data entry and data editing, and on the other hand the automatic processing of data and meta-data for use in other data processing software, e.g. for tabulation and analysis. In this way repeated data specification is no longer necessary, and consistency is enforced in all data processing steps. Figure 2 summarizes the integrated system for survey processing as it is realized at the Netherlands Central Bureau of Statistics (CBS) for most surveys.

Figure 2. Integrated Survey Processing



4. Computer assisted interviewing

While the traditional data collection was very much paper and pencil oriented, the modern way is Computer Assisted Interviewing (CAI). This allows for the integration of various traditional activities such as data collection, data entry and data editing, into one interactive cycle. This is most visible in case of CAPI (Computer Assisted Personal Interviewing) where the interviewing process, including routing and checking, is guided by a program in the laptop computer of the interviewer. But also in case of CATI (Computer Assisted Telephone Interviewing) and CASI (Computer Assisted Self Interviewing) this integration of traditional steps is crucial, resulting in a clean, machine readable record directly after the completion of the interview. If for some reason mail out/mail back is preferred (e.g. in case the respondent has to lookup some ledgers), the returned questionnaire can be entered and edited in one step with a CADI-program (Computer Assisted Data Input). This program provides an intelligent and interactive environment for data entry and data editing, of data collected by means of questionnaires on paper forms. The same holds for any PAPI (Paper and Pencil Interviewing) questionnaire. Whatever form of data collection is used, the result will be a 'clean' data file, i.e. a file in which no more errors can be detected.

Besides the interviewing process itself, the survey design has also changed considerable due to the introduction of new technologies. This is most visible in case of CAI. Typically, the design and testing of the interviewing program, including all the rules and edits, takes more time than in case of the paper-and-pencil interview. At the same time, this allows us to speed up the process after the data collection substantially, so that it is often a shift in time spent from after to before the field work.

The design of computer assisted interviews looks very much like the design of a traditional computer program. The tools used are so-called authoring programs, like CASES, AUTOQUEST or BLAISE, which allow the designer to control all aspects of the interview. In some way this resembles the design of an expert system: each questionnaire is defined by specifying "facts" like questions, answer categories and such on one hand, and "rules" to preserve consistency and a correct routing (skipping pattern) on the other hand.

Although CATI, CASI and CADI are carried out on desktop computers instead of on (portable) laptop computers as in case of CAPI, it is preferable to use similar software and hardware for all techniques. This allows for multimode surveys, where for example part is done by CATI and part by CAPI. Together with the requirements of portability, cost effectiveness and interactive use, this directly leads to microcomputers for all these activities, preferably connected at the office in a Local Area Network for easy data sharing. The advantage of having one cheap computer on the desk capable of handling all interactive tasks for each person separately, is so great that minicomputer-based systems for CATI or CADI definitely seem a way of the past.

The most advanced method is CASI (Computer Assisted Self Interviewing). The simplest form is where the respondent interacts with a computer program similar to the CAPI program for the interviewer. Since respondents can not be trained like interviewers clearly this method demands a very friendly interviewing program, in particular when the program is used without an interviewer around. The data resulting from the interview is then sent to the survey organization. This can be done either on floppy disk by the respondent or automatically by the interviewing program using electronic mail, as in case of the Telepanel system (see Saris, 1992) or the IRIS system at CBS. In order to provide some incentives to use this kind of automated interviewing, these programs often provide some coding and reporting facilities or even electronic games (as in case of the Telepanel) for the respondents.

Instead of PC's running CASI programs, some simpler alternatives are developed using the telephone. At the US Bureau of Labour Statistics (BLS), for example, touch-tone telephones and in the future voice recognition systems are used for simple surveys where the respondent only has to provide a few basic numbers like the monthly turnover. One of the advantages of this and other CASI programs is that the respondents can determine the optimal moment of "interviewing" themselves.

Ultimately, there is no interviewer or respondent left, resulting in a machine to machine interview. Especially for establishment surveys, this seems to be a promising future prospect. The keyword here is EDI (for Electronic Data Interchange). Using EDI, two or more computers can exchange information from corporate databases, according to very strict rules and protocols (like those in the so-called UN/EDIFACT settings). Of course, not only the communications and messages' protocols should be exactly defined, but also the items in the corporate database. This means that statistical and administrative concepts have to be standardized and harmonised, possibly by the statistical use of other public concepts, like fiscal ones. This will also substantially lower the respondent burden.

With many CAPI laptops in the field or dozens of interviewers doing concurrent CATI work, good CAPI and CATI case management systems are necessary. For an example of a CAPI case management system, including address-list management and multiple-survey management on the laptop, see Hofman and Keller (1991). For an advanced administrative and logistic system supplementing the CAPI panel survey of the British Labour Force, see Manners, 1992.

Several large national statistical agencies are using or planning to use techniques like CAPI or CATI; some are already in large scale production for some years while others are evaluating it seriously. See Statistics Sweden (1990), Braslins et al (1992), Nicholls and Matchett (1992) or Bethlehem et al. (1991).

5. Subsequent processing

After the interviewing and editing stage, several other activities are necessary. While the first stage can be viewed as the (often parallel) processing of individual records (e.g. one per CAPI session), the analysis step in survey processing often involves the whole batch of records. Something between the two worlds is the coding operation, where codes are assigned to descriptions, e.g. for type of occupation, branch of industry or education. Although coding is preferably be done automatically, in practice a substantial number of codes can not be assigned automatically and are therefore done interactively (as most of the coding for the Dutch Labour Force Survey at CBS).

Since complex coding operations often require large dictionaries or thesauruses, most of the coding work is usually done at the survey organization instead of in the field. However, simple code schemes (one or two digit coding for example) can be done very well on the laptop interactively, using menu hierarchies supplemented with simple word recognition systems, as in the Blaise coding module (see also Manners, 1992). Most coding algorithms focus on lexical or logical analysis, using already classified cases as a training set (see Lyberg and Dean, 1992). Other institutions like the US Census Bureau focuses more on simpler matching rules (i.e. nearest neighbour) but larger dictionaries (132 000 cases with descriptions on employer, industry, occupation and duties) in their experiments, using massive parallel computers for industry and occupation coding (see Creecy, 1992). Generalized systems for automatic coding are e.g. QUID (see Lorigny, 1988), developed at INSEE, France, and ACTR (see Wenzowski, 1988) from Statistics Canada.

Other steps in the analysis stage involve batch editing, imputation, post-stratification/weighting and tabulation. Although batch editing is less and less necessary for CAPI/CATI interviews (since all checks can often be done during the interview), most mail surveys may benefit from it. Some CADI systems support batchwise editing, preferably using the same or similar specifications as for the interactive CADI program so that everything can be specified once in the design stage. An additional advantage of this is that the CADI program can be used interactively after the batch to clean up the failed records.

An alternative to interactive editing can be found in fully automated edit and imputation programs, like GEIS from Statistics Canada, SPEER from the US Bureau of the Census, and the DIA system from the National Statistical Institute of Spain. The first two systems deal only with quantitative (mainly economic) data, while the DIA system focus on qualitative (often demographic) data. These programs implement linear or ratio edits, hot-deck imputation or the so-called Felligi and Holt approach. The editing and imputation process in these systems are mainly done in batch. For recent overviews on data editing and imputation, see SPWP (1990), Pierzchala (1990) and Kovar (1992).

The jury is still out on the case of batch versus interactive editing. While there is some evidence that batch is more cost effective in particular for large surveys like censuses, experience at CBS shows that interactive systems for economic surveys, like industrial production and foreign trade, can be successful. However, batch programs will seldomly need mainframes to do the job, and even large samples can be processed on super micro's and/or workstations these days (see, e.g. Quantime, 1991). Since so much of the editing and imputation involves rules and reasoning, I expect rule based systems to show up more frequently in the future.

Finally, after some weighting has been carried out, the results of the survey should be tabulated and published. As with many of the above steps, the programs involved should connect seamlessly to each other. Particularly, this is important for the treatment of meta-data (i.e. the information describing the characteristics of the data itself), like type, format, labelling, definition, etc.). One of the biggest advantages of most CAPI/CATI programs is that after the survey design stage it is seldomly necessary to respecify the data or the results of the data processing. Often the output of these programs can be used directly, without respecification, as the input for other analysis programs like SPSS or SAS.

Some systems even provide their own integrated suite of programs for nearly all the steps in survey processing: The Blaise system, for example, entails not only the CAPI/CATI/CADI authoring environment (including interactive coding and batch edit facilities) but also programs for data manipulation (called Manipula), weighting (Bascula) and tabulation (Abacus) in addition to set-up generators for other programs like SPSS (see figure 2). The backbone of this system is the dictionary or knowledge base, specified by the subject-matter specialist in a powerful, structured language that describes the questionnaire: questions, possible answers, routing, range checks and consistency checks. With the thus specified questionnaire as input, the system automatically generates the necessary set-ups for all further data processing. Since all these programs run at least at PC (LAN)-platforms, they are ideal for team wise, decentralized survey processing. See, e.g. Bethlehem et al. (1991) and Bethlehem and Keller (1992).

6. Publication

With a larger emphasis on our customers, more and more significance is put on timeliness of our statistics and on the attractive presentation of statistical results. One of the important things in the making of timely publications is the logistics of the survey processing. The more departments, specialists, cycles, and people are involved, the slower the production process will go. In general, the amount of time to finish up a task is more related to the number of departments involved than to the time per department. Therefore, decentralisation and team effort, integration of the various processing steps and standardisation of tools are all key elements of a fast process.

As critical the connectivity is between data collection, editing, and analysis (in terms of the exchange of data and meta-data), the same holds for the publication stage. Ideally, the output of the aggregation/ tabulation process should be available for distribution as fast as possible. The best way to do so is by making the results directly available on on-line databases for immediate access by customers. The same database can then be used for automatic document preparation (e.g. monthly statistical bulletins), preferably without any human interference (again: the more departments involved, the slower). And once all the data is in the database and well described and defined in its dictionary, it becomes easy to take snapshots out of the database and put them on floppy-disk or CD-ROM for further distribution. Products like PC-AXIS (Statistics Sweden), the Nord Rhein-Westphalen CD-ROM and CBSView (CBS) show the trend in user-friendly PC-based dissemination software.

Besides tabular information, micro data consisting of individual records per respondent are becoming more and more visible as a product of statistical agencies. At Statistics Canada the latter demand has given rise to an automated system called PASS (the Products and Services System for the Census of Population, see Hutton and Turner, 1992) which provide both dictionary services as well as ad-hoc tabulations. A final field of rising interest is graphical, in particular geographical data. After the great success of the US Census Bureau's TIGER tapes and CD-ROMS (with geographical information on the block level), some other agencies are starting to provide this kind of spatial data.

7. The future role of EDP in survey processing

In the past, the large computer was surrounded by a lot of people. In the future, however, the human individual in the focal point instead of the computer. The consequences are that in tomorrow's statistical agencies most (electronic) processing will be done decentralized (including management of the process and most of the development of new systems). Does this mean that everything should be decentralized? No, since all the things we want to be shared by several people need some co-ordination and therefore some centralisation. Thus, shared resources such as shared data and shared tools are to be "centralized" in some way or another. Shared data is important since changes by one person in the mutual data should be "seen" by other persons, which is impossible if we all have our private copy (this is the famous data integrity problem). Shared tools are important to guarantee data integrity on one hand and efficient processes on the other. Therefore, some standardisation of hardware, software and data communication is vital in order to facilitate the exchange of data, programs and experiences and the sharing of resources.

In the future, with most processing power on our desk instead of in the glass-house of the EDP department, an advanced architecture is needed to enable the sharing of resources. Here, the so-called Client/Server architecture will probably be the most dominant scheme in EDP. In this architecture, the user with its desktop (or "client") computer becomes the starting point, while the sharing of resources is accomplished by connecting these desktops with each other and with central databases on so-called servers. Data communication is expected to be crucial here: the Local or Wide Area Network (LAN or WAN) will be the backbone of future computing. Connecting a large number of clients and servers asks for strict standardisation and therefore of some centralisation. Here is an important role for the new EDP department. Since most present desktop computers are PC's (the absolute number one computer in the world with more than 100 million units sold), the biggest challenge for EDP is the co-ordination of all these (often stand-alone) desktop computers to work together in the future Client/Server architecture.

Introduction of this new approach demands more standardisation of the desktop than is visible in most organizations today, where the desktop is part of an "anarchy" called "end-user computing", totally out of control of EDP. There, also application development is split into two worlds: at the EDP department people are still oriented towards the old mainframe and 3rd-generation programming languages like COBOL, while the end-users prefer PC's, Windows and spreadsheets. With these two worlds drifting apart instead of coming together (as is needed for the true Client/Server architecture), a potential EDP crisis becomes visible.

Since traditional development cycles using 3rd generation tools and large computers are shown to be long and risky, EDP people tend to use long horizons for new systems. It is not unusual to discuss the introduction of new CAPI/CATI systems for certain surveys starting over 5 or more years. However, since the technology changes so fast these days (with more processing power, cheaper memories and better software packages becoming available each year), planning for these large horizons often means large problems. The alternative, buying someone else' software is also not an easy thing to do. Traditional risk-averse EDP people tend to come up with a long and extensive requirement-list for commercial CAPI/CATI software. Then, it is not surprisingly to hear that no products were found that meet all the requirements. These findings often resulted in the conclusion that internal development of the ideal system was the only way to go, sometimes leading to the next disappointment if development has taken longer than planned and proved to be more difficult than expected. The lesson learned from these experiences is that complete checklists will often result in complete stops. Buying software often turns out to be a better, although possibly not perfect, alternative in the end.

Even with commercial software, some internal software development is sometimes necessary. A good way to develop software is, in my opinion, to use the method of continuous improvements: first come up with version one that meets the most important requirements, and then work up to an improved version 2, etc. This method, which is similar to the way most end-users develop software, is often more controllable and effective than the traditional way of developing the ultimate comprehensive do-it-all system in one long cycle. Setting a short deadline of say a year for the first version often helps to get the priorities better in line. With the modern PC-based software tools, new technological and organizational insights based on reactions from the users on the first version can often be incorporated in the second version. In this way, the software life cycle becomes much shorter: we improve or throw away old systems while catching up with the changing requirements. And since most users nowadays prefer word-processing, e-mail, spreadsheets, graphics, etc. on their PC, integrating EDP solutions with the end-user desktop world is often a requirement. This holds in particular for CAPI/CATI programs, which by their interactive nature fit especially well on the PC-platform.

For administrative purposes (like large scale case management), the Client/Server architecture with PC based client applications, talking to shared data on central database-servers over networks, provide the best of both worlds. The conditio sine qua non for this environment is of course a strict standardisation of the hardware, software and data-communication components in the networked environment: centralize (standardize) the 'tools' and the data but decentralize the use of the tools and the processing of the data.

8. Conclusions

In concluding, I would like to stress the importance of Information Technology for our work: it is a so-called Critical Success Factor for official statistics. Using techniques like CAPI and CATI can improve the cost-effectiveness, quality and timeliness of most statistical processes. But IT is too important to be left to EDP specialists alone. Therefore, get EDP people and end-users working together in small teams, and ask for practical results within, say, one year. Practical results don't mean high-tech cathedrals: even with simple PC-based software, large gains in using CAPI/CATI and similar techniques are possible. Ultimately, never forget that we want to provide actual, reliable and cost-effective statistical information to our customers and that is all what it is about.

References

Bethlehem, J.G. and W.J. Keller, 1991, The BLAISE System for Integrated Survey Processing. Survey Methodology Journal.

Bethlehem, J.B., F.M. Kellenbach, and W.J. Keller, 1991, Computer Assisted Information Processing at the Netherlands Central Bureau of Statistics. Report, CBS, Voorburg.

Braslins, A., M.Coutts, R.Jamieson, B. Williams, 1992, The Building of an Integrated Collection Operation in Statistics Canada's Regional Offices: A Time for a Change. Annual Research Conference, US Census Bureau, Washington.

Creecy, R.H., 1992, Massive Parallel Computing and Automated Industry and Occupational Coding. Eurostat seminar on New Techniques and Technologies for Statistics, Bonn.

Hofman, L.P.M.B and W.J. Keller, 1991, Design and Management of Computer Assisted Interviews in the Netherlands. ASA 1991 Joint meetings, Atlanta

Hutton, F.E. and M. Turner, 1992, PASS - Products and Services System. Eurostat seminar on New Techniques and Technologies for Statistics, Bonn.

Inference Data Systems Ltd, 1992, Precision Data Classification System. Ariel, Sydney, Australia.

Kovar, J., 1992, Recent Experiences with Data Editing. CES Working Session on Statistical Data Editing, Washington

Lorigny, J., 1988, QUID, A General Automatic Coding Method. Survey Methodology, vol. 14, pp. 289-298.

Lyberg, L. and P. Dean, 1991, Automated Coding of Survey Responses: An International Review. CES Working Session on Statistical Data Editing, Washington.

Manners, T., 1992, New Developments in Computer Assisted Survey Methodology (CASM) for the British Labour Force Survey and other OPCS Surveys. Annual Research Conference, US Census Bureau, Washington.

Nicholls, W.L, S.D. Matchett, 1992, Computer-Assisted Survey Information Collection Issues at the Census Bureau. Annual Research Conference, US Census Bureau, Washington.

Pierzchala, M., 1990, A Review of the State of the Art in Automated Data Editing and Imputation. Journal of Official Statistics, vol. 6, pp. 355-377.

Quantime, 1992, Quantime's work on the LFS. Employment Statistics Working Party, Eurostat.

Saris, W.E., 1992, A Complete automated system for Information Processing in Family Budget Research. Report, University of Amsterdam.

Statistics Sweden, 1990, Computer Assisted Data Collection Production Tests in the Labour Force Surveys August 1989.

SPWP (Statistical Policy Working Paper 18), 1990. Data Editing in Federal Statistical Agencies. Prepared by the Subcommittee on Data Editing in Federal Statistical Agencies, Statistical Policy Office, USA.

Wenzowski, M.J., 1988, ACTR, A Generalized Automated Coding System. Survey Methodology, vol. 14, pp. 299-307.

LABOUR FORCE SURVEY AT THE HSCO

Ka'roly Kerepesi
Hungarian Central Statistical Office, Budapest, Hungary

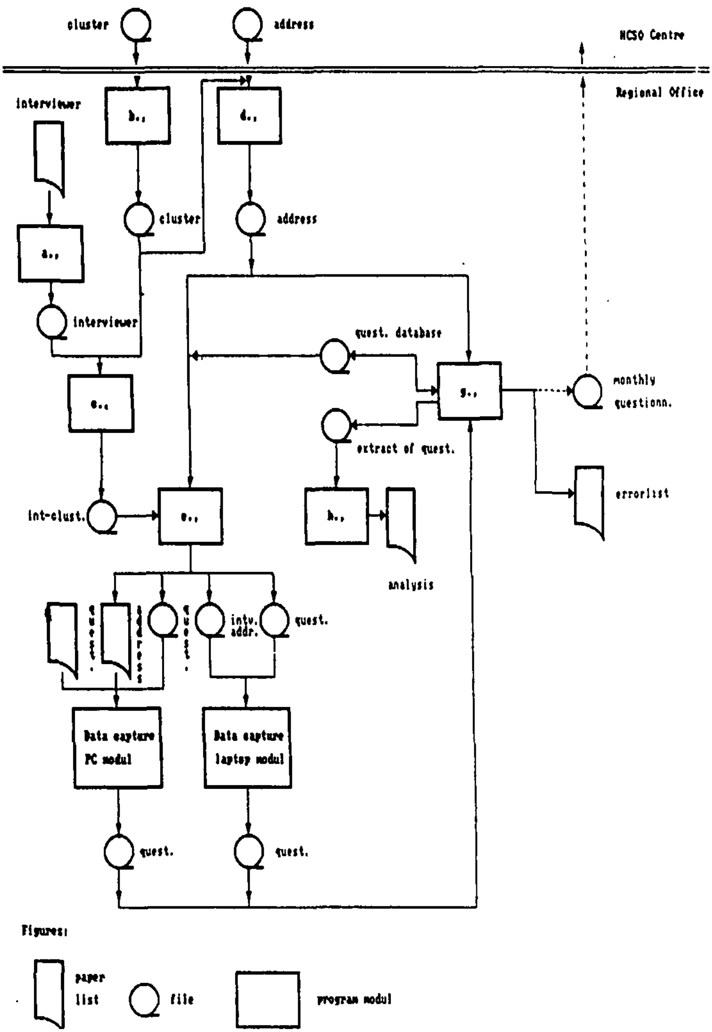
The new Labour Force Survey started at the Hungarian Central Statistical Office (HCSO) in January 1992. This study will describe the data collection and field management system of this survey.

The Labour Force is a quarterly survey. The sample consists of 30.000 households divided into 3 groups. These groups are visited monthly and the quarterly results are calculated from the monthly results. Every household is in the sample up to 6 months. Quarterly 1/6 of the households will be rotated from the sample and will be replaced by new ones. The addresses are taken from the Census address database. In the selection of the addresses we use our traditional cluster system.

The data collection is organized by our 20 regional offices. We have 600 interviewers taking part in this survey. 150 interviewers have laptop computers.

The data stream of the survey is shown in figure 1 on the next page.

Figure 1. Data stream of the HCSO Labour Force Survey



The functions of the system

1. Preparation
   a. Data of interviewers
   b. Regional cluster file handling
   c. Interviewer-cluster relation
   d. Address file handling
   e. Address and questionnaire disseminating based on c.

2. Data collection and capture
   PC and laptop programs

3. Field management control and analysis
   g. Completeness checking for addresses and questionnaire database update
   h. Field management analysis from questionnaire extracts

Our questionnaire consists of three parts. One household part, one or more persons and one or more económical activity parts (subfile structure). Because of the returning visits to the households we must handle the questionnaires. We decided to give them back to the capturing staff to overwrite them but before that we erased the information from the economical records (except the last occupation and workplace fields). We do not keep all the questionnaires in the database only the ones for the next quarter. The questionnaires of the rotated addresses are always deleted. So our questionnaire databases in the regional offices are compact: 2 to 6 MByte.

The Census address database is on a Novell-LAN in the HCSO Center in Budapest and maintained with Clipper programs. This database is an extract from the whole Census database. The department responsible for these data sends the addresses to the regional offices every month.

For data capture we use Blaise.

The field management software is written in Paradox and works on stand-alone PC's. On the laptops a C language program with the Paradox Engine makes the connection between the addresses and the Blaise questionnaires. The monthly addresses and the questionnaires of the previous interview are disseminated on floppy disks.

The reasons to use Paradox instead of Clipper were:
a. The graphical possibilities;
b. The Paradox Engine for our special 1 MByte disk laptop;
c. The SQL Link.
The field management software is disseminated with Paradox Runtime.

One of the main goals of making this system was to hand over most of the daily tasks (monthly preparation of the questionnaires, disseminating of the laptop questionnaire and address files) from the Computer Department to the Data Collection Department of the regional offices. These tasks were making utilities with Blaise (done before by the Computer Department). Now we can connect the interviewer administration with the data collection. The Data Collection Department has even the necessary tools to analyze the efficiency of the survey and the interviewers.

BLAISE IN SOCIAL SURVEYS AT THE NETHERLANDS CENTRAL BUREAU OF STATISTICS

Frans A.M. Kerssemakers
Netherlands Central Bureau of Statistics, Heerlen

1. History of computer-assisted interviewing at the CBS

The history of computer-assisted interviewing at the CBS dates back to 1981. At that time the CBS was considering the introduction of telephone interviewing, not at least because of the tantalizing promises of computer assistence which had shown up in various operational CATI-systems already. Around the middle of 1982, a ready-made package (ACRS, developed by M/A/R/C/ company) was installed and experiments were started to look into the feasibility of telepone interviewing. From 1984 computer-assisted interviewing has been in regular use at the CBS for telephone surveys with persons and households. With various revisions and adaptations the ACRS-package was used until 1990 when it was replaced by the CATI-machine of Blaise together with a closely connected, home-made management system for call scheduling, appointments and supervisory tasks. An accompanying change was the transition from a mini-computer (HP3000) to PC's in a local area network.

The first experiment with handheld computers took place in May 1984. The data collected were prices of commodities which had to be observed in shops. Around Christmas 1984 a comparitive survey was carried out to study the acceptance of computer-assisted personal interviewing (CAPI) by interviewers and respondents in a real face-to-face situation. Both experiments turned out to be a success. A major obstacle for more general applicability at that time, however, was that suitable and cost-efficient handheld computers could contain only one questionnaire whereas loading another questionnaire was cumbersome and could not be accomplished by the interviewers themselves. They had to be sent back to the CBS first.

It so happened, however, that in 1985 a project was started to build a continous Labour Force Survey. The number of interviews needed for this survey would comprise the larger part of all interviews with private persons at the CBS (each month, 10,000 addresses to assign to interviewers and data from about 10,000 persons). Besides, timeliness was one of the main objectives. If any survey, this one seemed to make the application of handhelds worth trying. After the software specialists succesfully succeeded to fit the rather extensive questionnaire in the computer a pilot study was carried out in March 1986 (van Bastelaer et al, 1988). A nice feature of the QUEST-program used then was that it kept record of the way the interviewer moved through the questionnaire and also recorded important information about how data were corrected. The problem how to get the data at the CBS (and get memory capacity available for fresh data) was solved by using modems for transmission by phone (the interviewer's handheld automatically phoning the host-computer at a pre-programmed time during the night). The test case showed no severe impediments for further implementation, except that the large-scale survey had to start on 1 January 1987 already. The main problem therefore became one of logistics, training and organization. In the meantime the original NEC CP/M-machine was replaced by the faster Epson PX-4 (also CP/M) with more memory capacity. New facilities were added to the home-made interviewing program and concurrently the questionnaire itself was significantly improved. Exclusively for the Labour Force Survey and with only minor adaptations, the whole CAPI-system was in full operation for five years, from 1987 till 1992 when it was replaced by the Blaise system. After some time the monthly sample became 12,000 addresses which were assigned to about 300 interviewers, each with their own Epson.

The third development which has had a tremendous impact on the collection and processing of data from social surveys at the CBS has been the implementation of the Blaise system. Blaise came in for the first time in 1987 when most surveys used paper-and-pencil questionnaires (except for the Labour Force Survey and some three CATI-surveys). Mainly as a result of an extensive data editing research project (Bethlehem, 1987) the CADI-machine of Blaise was introduced then to get rid of the time-consuming preparations of paper forms for data entry and to break through the subsequent cycle of computer checking and manual correction (for which paper forms often had to be consulted again). Instead, data entry and data editing of a paper form should be combined as much as possible in the hands of one specialist with the help of an intelligent data entry and error checking program. To this end CADI was used extensively from 1987. But then, after some two years, the (additional) provisions for the interviewing machine of Blaise were ready and a rapid shift occured to computer-assisted interviewing. Indispensable to this development was the availability of a MS-DOS machine wich could serve multi-survey purposes. Particularly the handling of different questionnaires by the interviewer, going from one address to another for different surveys, should not be too difficult. Once a suitable handheld with a disk drive for the exchange of questionnaires had been found (Toshiba T1000), three CAPI-surveys could already be conducted with Blaise in 1989 (apart from the Labour Force Survey which till 1992 used what was called QUEST, after the name of the questionnaire-interpreter).

Later on it was decided that from 1992 all interviewing for social surveys at the CBS, including panel surveys, should be carried out with either the CAPI- or the CATI-machine of Blaise. For special purposes, of course, paper questionnaires could still be used. With some minor exception (the small Housing Costs Survey) and with the Household Expenditure Survey still being in a process of conversion, this goal has almost been attained. From 1992 also, all CAPI-interviewers have been using the Toshiba T1000 SE with one disk drive and 384 KB RAM in addition to the common 640 KB work space.

2. Blaise-applications in the Department for Social Surveys

Within the CBS all data collection with persons and households has been centralized in one special department, i.e. the Department for Social Surveys. Here the fieldwork is prepared and organized, including the construction of the questionnaire and its conversion to Blaise. The interviewing is carried out by some 500 part-timers (for the most part housewives), nearly all of whom have their own handheld, and another 30 for telephone interviewing from the CBS-office at Heerlen. Besides pilot surveys, monthly samples contain 20,000 elements on an average, mainly addresses which have to be contacted. If only clear differences between questionnaires are taken as a criterion, the samples are spread over about 12 distinct kinds of surveys annually, not including specific screening actions with only a few questions. From the same viewpoint there have been 11 regular surveys which were predominantly CAPI or CATI until now and 4 which are going to be so in the short term (see table 1). In the more distant future other surveys are likely to come in, probably a Time Budget Survey and perhaps even a large Housing Survey with a sample size of nearly 100,000 (part of which could be handled using CBS-handhelds).

*Table 1. Regular social surveys at the CBS with CAPI/CATI (Blaise) (including in preparation for 1993, indicated by "**")*

Name of Survey	Panel survey	Gross sample in 1992, resp 1991 (*) or 1993 (**)	Predominantly CAPI	Predominantly CATI
1. Labour Force Survey		132,000	x	
2. Unemployed Labour Force Panel	x	12,000 (**)		x
3. Household Expenditure Survey	x	6,600 (**)	x	
4. Social Economic Panel	x	5,900	x	
5. Housing Costs Survey		1,100 (**)	x	
6. Consumer Sentiments Survey		22,500		x
7. Passenger Car Panel	x	6,720		x
8. National Travel Survey		18,600		x
9. Recreational Survey	x	25,200 (*)		x
10. Parliamentary Election Survey 1)	x	5,500	x	
11. School Career Surveys	x	7,250		x
12. Life Situation Survey		15,000	x	
13. Victimization Survey		6,000	x	
14. Health Survey		7,200	x	
15. Fertility Survey		20,000 (**)	x	

1) Conducted in 1989 with CAPI (Blaise) and now depending on whether there will be elections.

In the paper-and-pencil era the Department for Social Surveys also had to put a lot of labour intensive work into post-processing the collected data, both to make them machine-readable and to deliver 'clean' individual data records to the subject-matter departments. Now, it still has to take care of the quality of the collected data. But a far-reaching change has taken place as to the way in which and the stage at which the cleaning up is accomplished. If collected via CAPI/CATI data are already key-entried by the interviewer and so one can use computer files with the original data, thereby avoiding any discrepancy between what was registered by the interviewer and what is in the computer. Furthermore, a properly working interviewing machine will follow the routing instructions and only allow codes that are specified in the Blaise-questionnaire. The interviewer simply cannot violate these specifications and cause incomplete data in this respect. Apart from distortions by technical or operating failures the collected data will be 'syntactically' correct.

Whether they will make sense depends on other things too, particularly on the conceptual model for the specific subjects inquired, the design of the questionnaire, and the instruction and training of the interviewers. Perhaps the most important consequence of relatively clean data thanks to the Blaise-machine itself was that, because the bulk of clerical editing could get rid of, the idea was taking root to put data editing in the field all together, while at the same time reducing its necessity. Thus, attention was being focused on the interviewing situation and the questionnaire to a much larger extent and in a more systematic way than was usual when inconsistencies were dealt with in an extra round of data editing anyway.

Lacking a second chance, remaining inconsistencies within the range offered by the machine have been tackled since then by structuring the questionnaire so as to prevent errors from happening and by building both 'hard' and 'soft' checks at crucial points or where errors are likely to be made.

Moreover, there has been a growing awareness of the often inherent relation between the concepts and definitions which are used for a particular subject and the troublesome outcomes encountered. In the Labour Force Survey, for instance, there was considerable dissatisfaction with questions which were rather bluntly derived from administrative definitions, using terms which, although they sounded familiar, appeared to be rather fuzzy at the edges for many respondents, e.g. about status in employment. Also, an attempt to cover by retrospection the entire labour history since one year before the month of the interview failed in practice, being too demanding in many cases. These kind of experiences became the starting-point for a drastically revised questionnaire (in use now since January 1992). Many subject matters were removed or brought back to what really counted. At the same time the remaining concepts were elaborated in a much larger amount of simpler, more self-evident questions. The result was a very complex structure which really put Blaise to the test and would have been quite impossible using paper forms. But it has served to make the task of the interviewer (and the respondent) easier. Resources, like instructions, intermediate support and fieldwork monitoring can now be more effectively used for strategic issues, such as understanding the main structure of the questionnaire and considering the decisive factors at major branchings. This also greatly contributes to data quality.

Yet, in spite of all precautions data from the field will continue showing up inconsistencies, although less frequently and with less damaging effects. Again, an error detection program could be build to select 'dirty' records for interactive editing. But now it is more doubtful whether the extra information will outweigh the usually high costs, particularly if applied as a standard procedure within the social survey department. The fieldwork branch could add real information by going back to the interviewer or respondent but doing so on a large scale is hardly feasible in practice. Knowledge about the acting of interviewers and respondents can also be important for the interpretation of systematic errors. But as solutions can (also badly) affect the statistical outcomes it was decided that the subject matter specialists should control the adjustments of already collected data, also considering the use of alternative and statistically more sophisticated methods (e.g. those working on an aggregate level). Therefore data editing, if any, of data collected via CAPI/CATI should as a rule no longer take place in the social survey department. This, in fact, stresses the priority of the field and the questionnaire. As such, data checking for evaluative purposes is becoming more and more important, both to improve the questionnaires and to enhance the interviewers' performances. Besides, the mere savings as a result of drastically reducing the editing staff in the department make the use of laptops cost-effective.

Part of interactive data processing will remain at the Department for Social Surveys. It concerns special coding of some major socio-economic background variables. At present these are education, occupation and firm. When collecting information about these variables it is obligatory to use strictly prescribed question blocks (in Blaise), either a long or a short version. Because of the complex classification schemes and the predominant use of open questions coding demands specialized knowledge. To concentrate this knowledge and to further standardization of surveys from different departments coding specialists are working within the social survey department. The coding tasks are so specific that tailor-made programs are used. The resulting codes are usually written in the Blaise data files.

Now that paper forms are mainly applied in an additional way, for self-report booklets etcetera, the large-scale use of CADI has clearly diminished within the social survey department. Yet, apart from the (additional) paper forms, most CAPI- and CATI-surveys occasionally use CADI for limited tasks such as interactive coding of text answers, extra checks on data collected via CAPI or CATI, and afterwards repairing (the effects of) misspecifications in the questionnaire. Another application of CADI, mostly in other departments, can be found when a survey has different questionnaires and one wants to check on relations between them. Then the respective data files are often merged and after specifying a new structure CADI-facilities are used to accomplish this.

3. The embedding of Blaise in the fieldwork organization

To attain an efficient use of laptops Blaise-CAPI was first applied in two surveys for which fieldwork was carried out continuously. Most developmental efforts were aimed at building a functioning questionnaire. The operating task of the interviewer was thought to be simple. So there was no need for a master plan. The only thing the interviewer had to do before starting interviewing was to put the diskette in the disk drive and turn the laptop on, respectively turn it off after the last interview data were written to the diskette. Once the diskette came back there was some data manipulation, mostly by the programmer of the questionnaire, in order to prepare traditional data processing. In this manner Blaise-CAPI became rapidly implemented. Questionnaire design benefitted from this approach. Specialist knowledge of CAPI was quickly acquired, though with only a few persons. That the questionnaire program was open for manifold uses and reliable enough for further production was also confirmed. For all design problems a satisfactory solution could be found and no really serious errors were detected, not in Blaise itself nor in its application. But, this way of implementing Blaise-CAPI also laid bare the plain facts about what almost certainly would go wrong when more than about three surveys had to be dealt with. It came out that managing a multi-survey system was quite a step ahead of what had already been realized in the Labour Force Survey. First of all, even with only one diskette for each survey the interviewers seemed to have operating problems. Diskettes were taken out as they were written on. Interview data got lost as laptops were turned off before they were written to diskette (being no problem with the Epson where many were more familiar with, nor with the newer T1000 SE having a resume mode). Fortunately, total failures did not happen too often. Because all data from an interviewer and consequently nearly always from a particular area were on the diskette a backup should have been made anyhow. Especially worth noticing was that the actual cause of quite a lot of problems was panic after something unusual had came up. All these things indicated an obvious lack of control over the operating task. This again had it its effects on their clerical work, which, apart from the questionnaire, had not been altered. Sample addresses were still on paper, as were the fieldwork reports. The latter comprised a visit account and a sample account (there was a strict sampling scheme in case more households were found at an address). It turned out that interviewers were tending to become less accurate in writing (on forms) and entering (in the questionnaire) case identification numbers or filling in the accounting forms. This caused a lot of matching problems and, because of too many incomplete data, made the forms almost useless. Every project team tried to take its measures. But as these were hardly co-ordinated they embarrassed the interviewer, often making things even worse.

The description given is somewhat over-dramatic. For a lot of improvements could have been made by some co-ordinating measures and just tightening the reins. But it did work out that way and actually gave the impetus to what became the core of the CAPI management system. To take no further risks, as this was only the beginning, it was decided that interviewers should have their own computer-assisted management system, if possible. It should be a real help to the interviewers, enabling them to do what they had to do in an easy, smooth way and at a convenient moment, while concurrently checking on the completeness and consistency of their accounting reports. To familiarize and not scare them the latter should be left as much as possible as they were already. Of course, the address information should be in the laptop.

The result was a completely menu-driven system which was named LIPS (Laptop Information system for Personal Surveys). It was used for the first time in September 1990 and has been in full-scale use for all CAPI-surveys since January 1992. Following the routine procedure of a (female) interviewer visiting an address (which she has on paper also), she first selects from a menu on the screen the appropriate survey (after switching on the laptop, checking date and time and possibly being asked by LIPS to make a backup). She then selects the address from a list which can be sorted in different ways. Next she has to give some information about the address. In case there are sub-addresses (more specifically mail delivery points) she has to draw about half of these up to a maximum of three according to strict rules. At each address or sub-address she has to take all households, again up to a maximum of three. So a single sample address may be splitted up into nine sub-elements, which are represented on the screen as the elements of a 3x3 matrix. When she wants to interview a particular household she moves the pointer to the corresponding cell and enters 'i'. LIPS asks her to confirm her choice and after checking whether the correct questionnaire diskette is in the disk drive will start the Blaise-CAPI program. The (sub-)address identification number is automatically passed on to the questionnare and that is it.

After finishing interiewing she finds herself in LIPS again and enters 'v' to make her visit report. The system date and time are used as defaults and if she reports a response result LIPS will check whether a questionnaire was actually started (this and the questionnaire key are the only Blaise data LIPS knows about).

A well working system for data transmission is also essential, of course. Reducing peaks in work-loads and effectively monitoring fieldwork progress and interviewer performance require a steady flow of data. For interviewer-dependent data, transmission by telephone best suits the needs. As with the Labour Force Survey the laptop takes the initiative, but if there are any data to be sent to the interviewer the same session is used for this purpose. Of course, there can now be sent data for different surveys simultaneously (up to 15). The relatively huge programs for running the questionnaire are sent on diskette. They are on diskette in LIPS also and as they are same for all interviewers can easily be sent in the same way as the explanatory letter that usually accompanies a new questionnaire. Here, the relevant data are the data that can vary between interviewers.

To keep things manageable, also for the interviewer, this kind of information should not be too fragmentarily transmitted. Therefore all of the information pertaining to a particular address is packaged. When an interviewer decides to send an address back, all of it is put ready for sending, but LIPS will check on the completeness of the required address information. If something is missing the address can only be transmitted after being explicitly marked by the interviewer and in that case it will get special attention at the CBS. The same applies to an address that was not used at all. By giving it a different mark it can be quickly selected for re-assignment.

Thus, LIPS is a real case management system. It nicely fits in with the centralized case management system in the interview administration section at the CBS. The task of the latter is to assign sample cases to interviewers and to keep track of the progress made with respect to the handling of cases, mainly by keeping count of the results in reports about finished cases. These tasks have also been highly automated in a system called IAS (Interview Administration System). The assignments are first carried out automatically and then mostly manually adjusted. Thanks to LIPS and also the transmission by telephone the administration of cases has been greatly simplified. Cases can be easily traced and they can be ascribed a clear status, even though LIPS is the domain of the interviewer. Closely connected to IAS is also an automated Interview Declaration System (IDS).

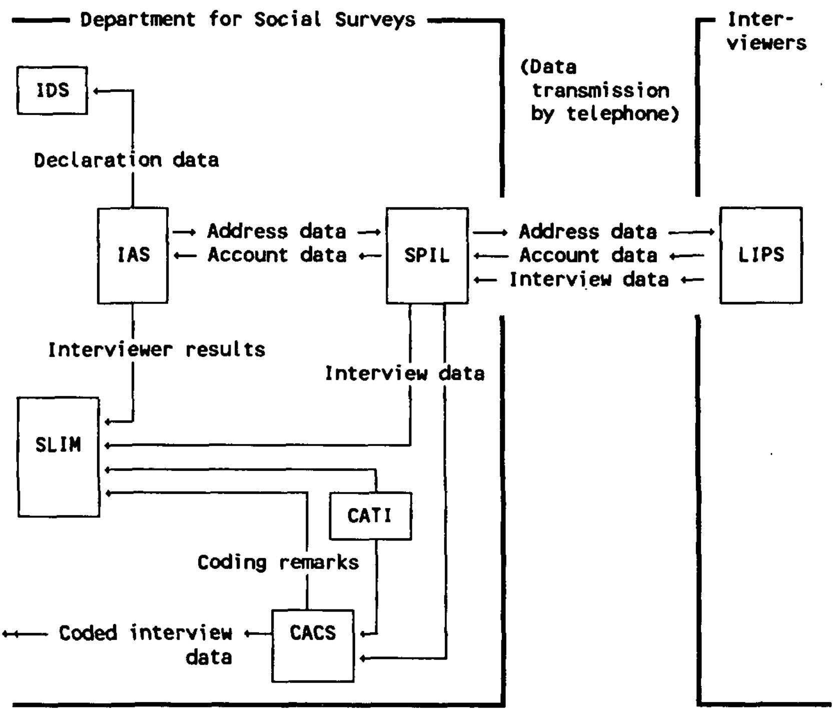
Finally, SPIL should be mentioned, being the System to Perform data-Interchange with Laptops (why not with LIPS ?). It arranges everything that is necessary to transmit data to the field, such as making interviewer packages and compressing the data. It also distributes the received data from the field, putting them in the right form. Normally the administrative data go to IAS and the statistical data to CACS for special coding (see section 2).

There is a lot more to tell about SPIL, LIPS and features of the data transmission. But these are described elsewhere (Hofman et al, 1991a). Being far from simple, the building of the systems required a great deal of effort, a tour de force indeed, but the result has been highly rewarding. Looking back, one could say that the start of the CAPI implementation was not so unhappy after all, at least not in its consequences.

The price that had to be paid was extensive interviewer instruction, training and support. The interviewer not only had to learn how to deal with LIPS and with data communication, but also had to understand the place of LIPS in the survey organization, being mainly a prolongation of the case management system. The most difficult part was the handling of the different diskettes. The interviewer should grasp basic notions like the difference between the questionnaire and the answer data, or between internal and external memory. An intensive instruction of three days was necessary. Besides, fake addresses and a normal questionnaire have been used to exercise in LIPS.

More on a meta-level and still developing is the System for Laptop Interview Monitoring or SLIM (Dutch for 'clever'). It is being built and partially used already to serve the fieldwork supervisors. It should become a general system for quality control as far as the fieldwork is concerned.

Figure 1. Automated systems for data collection and data processing



The main problems during the implementation of the Blaise-CATI system arose from not being familiar with the many options for call management that are offered by the system. Because of the freedom for the user it is easy to create huge planning problems (too few or too many interviewers in a shift). By learning how to tune the option parameters the problems gradually disappeared. Interesting about the system is that both the call management data and the appointment data are made part of the Blaise-questionnaire. It turned out that this integration in Blaise had many advantages because all Blaise tools for data manipulation could be used (see Hofman et al, 1991b). The system is relatively simple, flexible and efficient.

Literature

van Bastelaer, A.M.L., F.A.M. Kerssemakers, D. Sikkel, 1988, Data Collection With Hand-held Computers: Contributions to Questionnaire Design. Journal of Official Statistics, vol.4, no.2, pp. 141-154.

Bethlehem, J.G., 1987, The Data Editing Research Project, in: Select 4, Automation of Survey Processing. CBS, Voorburg.

Hofman, L.P.M.B. and W.J. Keller, 1991a, LIPS, a Management System for Computer Assisted Interviewing, in: Netherlands Official Statistics, vol.6, no.3. CBS, Voorburg.

Hofman, L.P.M.B. and W.J. Keller, 1991b, Design and Management of Computer Assisted Interviews in The Netherlands, in: Computer Assisted Survey Information Collection: International Progress, American Statistcal Association, 1991 Joint Statistical Meetings. Atlanta, Georgia.

A PROTOTYPE SYSTEM OF DATA EXCHANGE FOR STATISTICS FINLAND'S CAPI SYSTEM

Vesa Kuusela
Statistics Finland, Helsinki, Finland and
Antti Merisalo
Tele-Mail Software Ltd, Helsinki ,Finland

1. Introduction

At Statistics Finland, Computer Aided Data Gathering (CADAG) has been under development for some time. In the first phase, a CATI centre was set up in the spring of 1991. Blaise was selected as the CADAG software. The first study was undertaken in May 1991, and since then five studies have been completed by CATI.

The second phase of the CADAG project was to analyze and to test various possibilities for putting the CAPI organization into operation. At the moment, CAPI activity is still in the planning stage. Statistics Finland has some 160 field interviewers and their central unit is in Helsinki where all of the interviewing is supervised. Only ten of the interviewers have a laptop computer but all interviewers will later be equipped with one.

It became apparent in the very beginning that the exchange of data between interviewers and the central unit was perhaps the most complicated part of a CAPI system. However, it is of major importance and needs to be designed carefully . If the subsystem for data exchange does not function adequately, it both ties up a lot of personnel and is error prone. In the worst case, all the benefits which were behind the decision to establish a CAPI system will be lost.

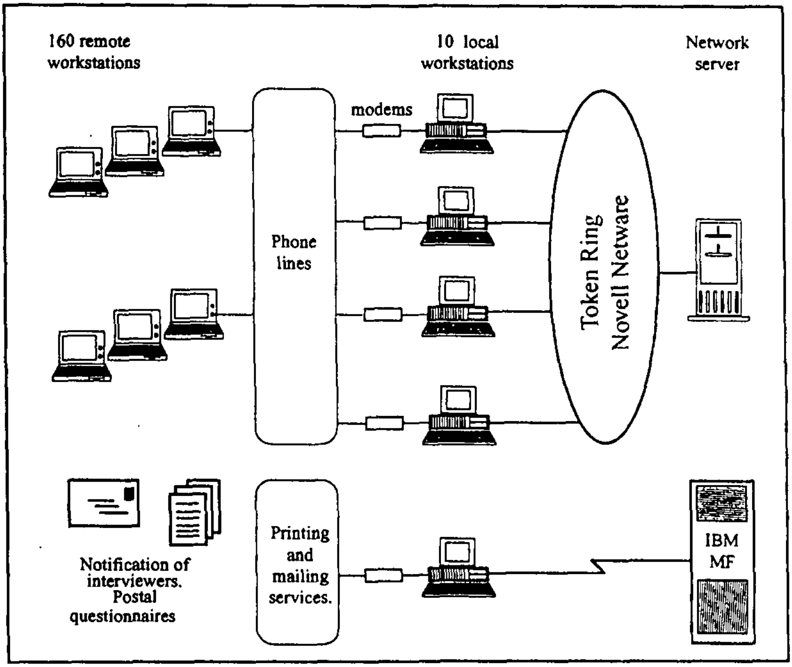
In this paper we describe the prototype of a system designed to enable the interviewers to receive questionnaires and contact information and to return data to the central unit.

2. Hardware

The basic idea behind the system is to use the CATI centre as the centre of services for CAPI as well. At the moment, the CATI centre has ten work stations connected to an i486-based server via Novell Netware and Token Ring. The CATI centre is in use mainly in the afternoon and in the evening. It is seldom used in the morning because many of the people to be interviewed are hard to reach at that time.

Because the work stations are in the CATI centre, there is naturally a telephone line for each of them. The CATI work stations needed only an internal 2400 baud modem to work as service stations. In addition, the modems can be used for automatic dialling, as well, when the feature is added to Blaise. Each work station can be used either in the CAPI service mode (and the modem is programmed to answer incoming telephone calls) or in the CATI interview station mode. This means that there are 1-10 telephone numbers available for field interviewers (see figure 1). The number of service stations is easy to increase when the CATI centre is enlarged. However, the current number of service stations probably will suffice even when all interviewers have laptops.

Figure 1. The hardware configuration of the CATI centre



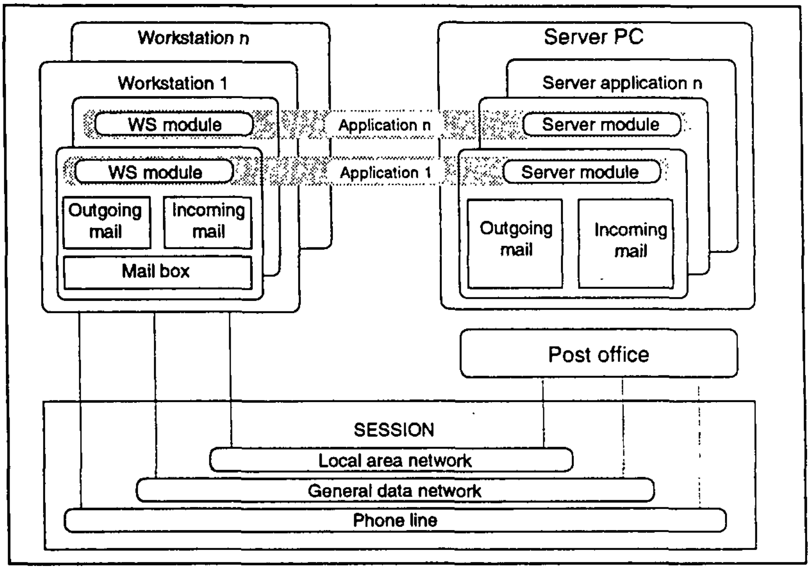
As the CAPI system is still in the planning and testing phase, only ten of the 160 field interviewers have a laptop computer at the moment. The computers are Sharp 4721's with a 20 Mb hard disk and a 1.44 Mb floppy drive. Interviewers also have an external 2400 baud modem. In addition, there are also some parts of Blaise and Turbo Pascal in each machine. The reason for this will be explained later .

3. Software

The software for data exchange was specially designed and made for this purpose by one of the authors (AM). Additionally some public domain software to pack and to transmit data are applied.

One design principle has been that the interviewers contact the central unit when it best suits them within a given time interval. Another principle has been that they need only a few easy commands despite of the fact that the processing capacity of their laptops is used to make the working questionnaire for the study. Only the text (ASCII) form of the questionnaire is sent to interviewers and it is compiled in their machines. This is done in a batch run.

Figure 2. General description of the software system

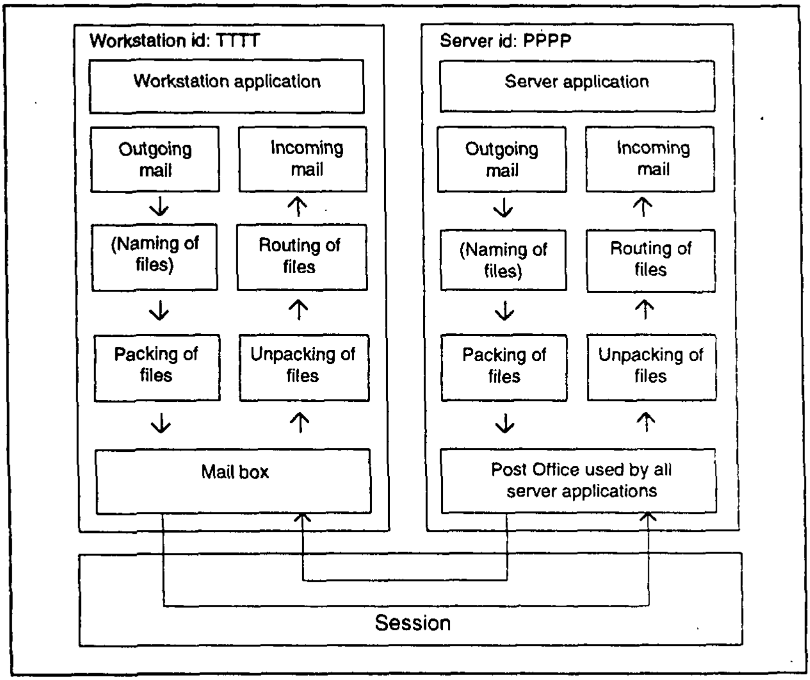


The system is described in figure 2. Every work station may have several CAPI applications at the same time. The server software may be in any machine connected to the LAN of the CATI centre. The machines in the CATI centre may be used both as servers for CAPI and as workstations for CATI.

4. Server Software

The server software is the part of the system which resides in the work stations of the CAPI service centre. The software may be in sending or in receiving mode during the same session. The switching between modes is automatic. A more detailed description of the logic and of the data flow in the system is given in the figure 3.

Figure 3. Detailed description of the data flow in the software system



When in sending mode, the server software prepares questionnaires (text files) to be sent, makes a packet for transmission, furnishes the packet with an address and time-stamp and moves the packet into the box for outbound post in the post office.

When in receiving mode, the server software unpacks the received data and moves them to the directory of the specific interviewer (all interviewers have a directory of their own). This is based on the name of the incoming file. The name is formed by combining the identification code of the interviewer and of the project code of the study.

Additionally, the server software maintains a log of transactions in the system. On some occasions, the server software joins together the separate interviewer data files of a study and moves the files further to be analyzed.

5. Work station software

The work station software resides in the interviewers laptop computers. It receives the packed questionnaire file, unpacks it and checks its integrity, makes a directory for the study and moves the unpacked questionnaire into it and compiles the questionnaire.

When the interviews of a particular study are finished, the interviewer sends the corresponding data files (*.D?? files) back to the CAPI service centre by a single command.

The work station software and the server software communicate with each other and the interviewer does not have to do anything else except invoke the system. Even dialling and the change of the telephone number in case the line was busy is done by the software.

6. Organization of the field work

Contact information (i.e. the respondents' names, addresses, telephone numbers etc.) will be sent to the interviewers by mail. This is the signal for an interviewer to contact the central unit via modem in order to copy the questionnaire from the centre to his/hers laptop. If interviewers have data to send back it will be done automatically at the same session. When an interviewer has received the questionnaire, a batch run will start. The batch run contains commands to check the questionnaire's syntax and to compile the questionnaire.

When the field work of a study has been completed, the interviewer sends the collected data files back to the central unit. The server has a directory for each interviewer where the software takes the received files.

7. Summary

The CAPI organization at Statistics Finland is partly based on the use of the CATI centre as a service centre for data exchange between interviewers and the central unit. The system is still a prototype with limited testing possibilities but the results of the few tests have been promising. Compared to many other potential solutions, the prototype described above has the following advantages:

1. The volume of data exchanged by modem is kept to a minimum. With 2400 baud modems, the contact times grow rapidly if the amount of transmitted data grows. Long contact times generate large telephone bills, and they are more error prone and inconvenient.

2. The utilization of the CATI centre grows. It is not very rewarding to try to reach people before afternoon. Therefore the first shift usually starts in the afternoon and the CATI centre is not used before that. The morning is a suitable time for data exchange with the field interviewers.

3. The system is inexpensive. The only additional investments are the modems in the CATI centre and the software. Standard modems are inexpensive and may be used in the CATI centre in the future when automatic dialling is added to Blaise.

4. Easy to expand. If the system faces traffic jams it may be easily enlarged by new modems.

5. The system may be later expanded as an office system for interviewers.

THE USE OF BLAISE FOR ADMINISTRATIVE PURPOSES

Johan P.M. Lammers
Netherlands Central Bureau of Statistics, Heerlen

In the past years most of the applications built with Blaise were of a statistical nature. This is not surprising regarding the goals for the development of Blaise. Most of the checks and corrections on the collected statistical information had to take place interactively and simultaneously while entering the relevant data. A large number of the checks can actually be executed this way by Blaise.

However, the process of collecting, checking and correcting statistical data comprises more than just processing the collected data. Efficient processing requires some kind of administrative support system. Such a system can improve both the data collection proces and the quality of the collected information. Vital for the whole statistical production process is a close and good link between the statistical system (usually implemented in Blaise) and the administrative system.

A link between a statistical system and an administrative system can be realized in several ways. Depending on the information needs of both systems, one can choose between:

1. Linking based on batch-wise processing: All data is processed in batch by one system, and then all data is processed in batch by the other system. This way of working offers the possibility of checking the data and printing the errors on error-lists. Like in the past, these lists will then be used to correct the relevant data.

2. Linking based on both batch-wise and interactive processing: the data in one of the systems is frozen while the other process can interactively use these frozen data for checking and correcting.

3. Linking based on interactive processing: both systems can interactively use each others information. This approach has the advantage that the information is always up to date.

Blaise 2.x offered the possibilty to realize each of the above-mentioned approaches to linking systems. The second approach requires an external data file while for the third approach there is a necessity that both processes make use of Blaise (and can use each other as an external Blaise file). This aspect causes a new problem: is Blaise suitable to implement administrative systems? In order to judge this suitability, an inventory is required of the characteristics of administrative applications, the possibilities of Blaise to support those characteristics and, if possible, the experience in this area. The rest of this paper will deal with these topics.

The administrative system must register for each unit to be observed the source of information, the way the information is collected, and the result(s) of the collection attempt. This general description covers a large variety of collection strategies. Depending on what is possible in a specific survey, use is made face-to-face, telephone or mail interviewing, while it is also not uncommon to use secondary information.

The way of data collection is also reflected in the characteristics of the administration system that is implemented. In the context of this paper I will not discuss these differences, but I will restrict myself to general characteristics of such systems that are required, whatever way of data collection is used. In this respect, administrative systems can be characterized by:

1. A large degree of repetition due to the fact that activities concerning a single unit are carried out at varous moments in time. It means that the data concerning a unit will be changed often and in different ways. Statistical systems do not have this characteristics. In principle, a unit in a statistical system will be manipulated only once. The data are checked and corrected and then they are ready. Blaise is developed for this kind of activities. Administrative manipulations are necessary:

- to define the way in which the information is to be collected (how to do it and where to do it);
- to indicate the start of collection phase;
- to indicate the end of the collection phase;
- to manage recalls;
- to make changes in the information required for the collection phase (new adresses, telephone numbers, etc);
- to re-define the way in which the information is to collected (if a previous method failed);
- the nonresponse administration, etcetera.

Each of these manipulations only concern a limited set of items in the administration, while other items always remain unchanged (they are 'read-only').

2. Manipulations should be carried out in a quick and simple way. The checks on these manipulations are simple but they vary by type of manipulation (see also 1.). Each type of manipulation requires an adequate protection concerning the set of items that has to be changed and to set of data that cannot be changed. Part of the manipulations can be carried out using batch-processes because all the relevant information is already available within the system.

Administrative treatment of information differs from statistical treatment in several respects. The largest problem arising regarding the suitability of Blaise for administrative treatment, is the combination of checks concerning the different types of manipulation and the desired protection of the data. Ideally, every type of manipulation should have its own routing. However, Blaise supports only one route, and that is the route used to get the answers to the questions in the questionnaire.

Another property of Blaise is that answers to questions that are on the route are set to 'empty'. From a statistical view this is correct. In administrative applications there may be different routes for different manipulations, but changing the route structure may not affect the answers to questions not on the route. Therefore, Blaise seems less suitable for this kind of activity.

Still, possibilities exist to solve this problem using a less obvious implementation of Blaise. This implementation contains not one but two sets of related questions. The first set contains questions with the attribute SCREEN and serves the interactive treatment of different types of manipulations. The second set contains the attribute HIDDEN in order to save all the important data, regardless the routing of the "screen"-questions. This second set is part of every possible routing. The check paragraph specifies the relationships between both sets.

In order to illustrate the previously mentioned technique, a simple sample Blaise questionnaire will be described in which two different types of administrative treatment can be carried out.

```
QUESTION ADMISYMP "Administrative application, Blaise symposium";

QUEST
   Id            : STRING[1] (KEY);
   H_Mail_Out    : DATETYPE  (HIDDEN);
   H_Mail_Back   : DATETYPE  (HIDDEN);
   Manipul   "//Mail out date : $H_Mail_Out //Mail back date: $H_Mail_Back////"
                 : (Mail_out "Mail out", Mail_back "Mail back") (SCREEN);
   S_Mail_Out    : DATETYPE (SCREEN);
   S_Mail_Back   : DATETYPE (SCREEN);

ROUTE
   Id; Manipul;
   IF Manipul = Mail_Out  THEN S_Mail_Out  ENDIF;
   IF Manipul = Mail_Back THEN S_Mail_Back ENDIF;
   H_Mail_Out; H_Mail_Back;

CHECK
   IF S_Mail_Out = EMPTY THEN
      COMPUTE S_Mail_Out:= H_Mail_Out;
   ENDIF;
   IF S_Mail_Back = EMPTY THEN
      COMPUTE S_Mail_Back:= H_Mail_Back;
   ENDIF;
   IF (H_Mail_Out <> S_Mail_Out) AND (S_Mail_Out <> EMPTY) THEN
      COMPUTE H_Mail_Out:= S_Mail_Out;
   ENDIF;
   IF (H_Mail_Back <> S_Mail_Back) AND (S_Mail_Back <> EMPTY) THEN
      COMPUTE H_Mail_Back:= S_Mail_Back;
   ENDIF;

ENDQUEST.
```

This questionnaire will only be effective if it is turned into a CAPI- or CATI-program, because they apply active routing. A CADI program will always produce routing errors.

The desired functionality may be achieved in this way! For each type of manipulation a specific route is possible, and at the same time the administrative information is saved and is ready to be read whenever it's necessary. The administrative information is available in part of the questionnaire by means of the set of hidden questions.

An elegant way to show this information is to include the hidden questions in the question text of a central question to guide the different routes (in this example the hidden questions are included in the question: Manipul). Each type of manipulation starts with this question and in most of the error situations Blaise will by default jump to this question.

Furthermore, a very important aspect of administration is the use of the registered data. Except for checking and correcting purposes in single cases, the information can also be useful in several other situations, for example reports about the progress of the collection process, lists of units not yet completed, information for recalls, general management information, etc. An administrative application that uses Blaise should also be able to produce these kinds of information. The tools that accompany Blaise, Abacus and Manipula, offer these possibilities, because they operate directly on one or more Blaise data files, index-sequential files and/or ASCII files, and where selection, projection and combination of information is possible.

The department of "Economic Censuses" has experienced with the idea of administrative systems during the last 2 years. The statistical applications where accompanied by two administrative applications (a general registration and a specific registration of interview visits to companies by CBS employees). These applications were linked directly as external Blaise data files. Batch manipulations were implemented in Manipula. This resulted in convertable ASCII files. A subsequent ASCII-Blaise conversion made those transactions complete.

For the surveys of the department, various data collection techniques are used. The department took care of maintenance of all these applications. This maintenance consisted of adjustment of external paragraphs for all the different statistical applications, the adjustment of programs that realised the manipulations in batch, and the adjustment of programs that generate progress reports (a different collection strategy often requires a different progress report). The maintenance could be realised relatively simple and in a short period.

Concerning data security, this approach offered the possibility to separate Blaise-applications, and to let management of the parts be carried out by those organisational units that carry responsability for the relevant part of the data. All the security demands can be taken care off at the technical level of the operating system (MS-DOS/Novell), by setting the proper rights.

A problem was the sensitivity of Blaise for disturbances on the hardware-level and for incorrect user interrupts. In my opinion, this general problem becomes more serious when more and more Blaise applications are linked in the above mentioned way, because other applications do not work any more when one of the applications involved has a disturbance.

Summarizing one can say that Blaise can be used to implement administrative applications. The high degree of exchangebility of information between administrative and statistical applications offers a range of possibilities to use each others data. The administrative applications can be maintained well although the construction seems a bit artificial. This results in a flexible application. Security aspects on the logic level are no problem when the operating system can handle them. On the physical level however the security has to improve.

THE CHOICE OF BLAISE IN THE FAMILY RESOURCES SURVEY

Stuart Mitchenall and Rory MacNeill
SIA Limited

1. Background

The Department of Social Security (DSS) has responsibility within the UK for the payment of benefits and the collection of contributions under the National Insurance and Industrial Injuries schemes; for the payment of child benefit and one parent benefit; and, on a means-tested basis, for paying (as an example) Income Support and Family Credit. It also administers the Social Fund and pays a number of non-contributory benefits which are non means tested.

In the period from April 1992 to March 1993 it is estimated that the DSS will deliver over £70 billion of benefits of which £26 billion is non-contributory benefit and thus related to circumstances rather than contributions.

It is essential that the DSS has the ability to analyse client behaviour through the availability of accurate, up to date management information. Data is required on the eligibility for benefit as well as the actual take-up of benefits.

To date the DSS has used data collected in the Family Expenditure Survey (FES) for this purpose. This, however, is considered to be inadequate as the data is collected from too small a sample for DSS purposes and it is not available until long after the date of interview. The DSS believes that its analytical powers will be increased through the availability of a new data source under the control of the DSS.

The Family Resources Survey (FRS) was launched on 1st October 1992 to meet the DSS requirements. This launch was preceded by over 2 years preparation, design and piloting. The aim is to achieve 25,000 successful interviews from a sample of 40,000 households. Each interview will include all members of the household and is expected to take approximately 90 minutes on average.

2. CAPI

Early in the development of the FRS it was suggested that the use of Computer Assisted Personal Interviewing (CAPI) in the survey could be advantageous. At that time (May 1990) there was little experience of CAPI in the project team and no knowledge of any application approaching the size and complexity of the FRS.

However, the concept of CAPI was recognised as being in line with the objectives of the FRS:

i) Quality
High quality data is desired. The interview is recognised as being very complex with a significant amount of financial data. The ability to carry out data validation at the point of capture is very attractive.

ii) Speed and timeliness
CAPI offered the potential for data analysis within timescales not possible using traditional methods.

iii) Costs
Overall costs could be reduced significantly using CAPI. The FRS was envisaged as being a long term project which would enable initial capital costs to be recouped by cheaper running costs.

The potential was considered great enough to spend time investigating the feasibility of using CAPI in the FRS: considering software, hardware, interviewer considerations and user experiences. An initial report by SIA Ltd recommended that CAPI was feasible if a suitable software package could be found. A trawl of the software market identified a number of packages but few that were sophisticated enough to handle the FRS. Indeed three packages were considered to be contenders : BLAISE, MICROTAB and QUANCEPT. The DSS agreed with the recommendation recognising that if the FRS was not launched using CAPI it would only be a short while until it would have to be transferred to CAPI.

3. Software evaluation

Accepting that CAPI was feasible if software was available that could handle the complexities of the FRS, the DSS commissioned a comprehensive evaluation of the leading software packages designed to answer whether any of the packages were capable of handling the survey and, if so, whether one package could be recommended as particularly suitable for this application.

The evaluation was carried out by SCPR with support from SIA Ltd, OPCS and Toshiba. The basic methodology adopted was to develop a prototype of the FRS in each of the three packages. These prototypes were then used to test the software against a set of identified evaluation criteria including the ability to:

- accept details of selected addresses from computerised source;
- record a number of visits made to a household;
- cope with concurrent interviewing of several members of a household;
- 'loop' through questions a set number of times defined by a previous response eg. variable number of children, bank accounts, jobs etc;
- amend the number of iterations of these loops during the interview;
- handle all standard types of question eg. single response, multiple response, open response, dates, monetary values (with decimal places) and percentages;

- incorporate text from previous questions in question text and code lists;

- deal with complex routing and routing implications of changing data;

- check ranges and ineligible formats;

- use 'hard' and 'soft' checks;

- interview several adults concurrently;

- record and calculate the interview length;

- code open responses following the interview.

Also of importance were the ease-of-use for interviewers; the capacity of the software in terms of numbers of questions; the ability to interface to other software for analysis and the support given by the software's distributors.

The software evaluation was carried out using approximately six man-months of researchers and programmers time. Additionally, significant support was given by the software suppliers and a small "field-test" was carried out using experienced interviewers.

The conclusion of the evaluation was that only one package - Blaise - could be used for the successive implementation of the FRS. A summary of the main findings is:

Software	Strengths	Weaknesses
BLAISE	'Table' format enabling many data items to be displayed on the screen	Inability to compile whole questionnaire in a single program
	General ease of use	Concerns over user support
	Robustness when answers are changed	
	Ability to handle real numbers and real arithmetic	
	Wide range of output formats	
MICROTAB	Quality of customer support	Doubts about ability to recover data after making amendments
	Efficient handling and storage of hierarchical data	Relative difficulty of moving around within the interview
	Facilities for summary screens	Restriction on the size of the serial number
QUANCEPT	None	Insufficient facilities for handling and manipulating numeric data
		Unable to handle questionnaire of desired length

4. Hardware and communications

Following the decision to proceed with CAPI using Blaise software the DSS was keen to ensure that maximum benefit was gained from the investment in technology. Each interviewer was to be equipped with a computer and was essential that the appropriate hardware was chosen for the FRS interview and also for any other associated tasks.

A number of detailed logistics concerning the running of the survey were dependent upon the field agency selected to carry out the interviewing which at this time was not known. However, it was important that in the planning and budgeting of the project that the DSS was aware of technical options and considerations.

A thorough review of the technical issues surrounding the decision to use CAPI was commissioned, in particular looking at:

i) Functions that could be carried out in the field using the PC.

ii) Data Communications.

iii) Hardware Specification.

iv) Approximate Costs.

A report on this review concluded:

i) Obviously the main function of the PC in the field is to carry out the interview itself but it was recognised that in addition to this it would be feasible, if desired, to:

a) Receive sample addresses directly from the survey managers via telephone lines;

b) Transmit program updates similarly;

c) Edit and code the data collected on the PC following interview;

d) Transmit data via telephone lines to a central database;

e) Record and transmit other data such as pay and expense claims.

ii) The interviewers should all have a modem for data communications; Either internal or external modems could be used. Internal modems have the advantage of always being with the PC should they be transferred between interviewers, external modems cause less of the drain on battery power.

The modems should meet industry standards MNP4 and MNP5 for error correction and data compression but it is not important that they have fast transmission speeds as relatively small amounts of data will be transferred.

At the point of data receipt similar modems are required. It was recommended that these modems be linked to a sophisticated modem management system and controller to log and check all data transfer.

iii) Initially the choice of hardware was between laptop and palmtop machines. It was quickly recognised that laptop machines were necessary for reasons of ease-of-use particularly with regard to keyboard size.

Having chosen laptop computers it was decided by the DSS to use Toshiba products. There are particular advantages in Toshiba equipment with regard to use in a CAPI environment - the machines can be left 'switched-on' between interviews - and also being one of the market leaders in portable equipment the DSS could be assured of the quality and service they desired.

Various specifications of PC were considered. It would be possible to run the application on an 8086 - based floppy drive machine (ie. a low specification) but there were some considerations:

i) The survey program would eventually be very large and the faster 286-based machines may be advantageous.

ii) Because the machine could be used for a number of functions a hard disk would be useful and would remove a number of potential problems arising from using floppy disks only.

iii) Hardware suppliers are no longer manufacturing low specification machines and it was recognised as a potential risk to choose a machine that may be discontinued early in the project.

Because of these considerations it was decided to recommend to the eventual research company that the machines to be used should be Toshiba, 286-based with a 20 Mbyte hard disk.

iv) A cost comparison between running the FRS using CAPI and traditional methods was carried out. This showed that the savings made in printing, postage, data preparation and administration more than covered the initial outlay on hardware, software and training after three years.

COMPUTER ASSISTED SURVEY METHODS (CASM) AT OPCS AND SOME CURRENT ISSUES IN THE USE OF BLAISE FOR THE LABOUR FORCE SURVEY

Tony Manners and Nicola Bennet
Office of Population Censuses and Surveys (OPCS), England

1. Introduction

This paper aims to explain briefly the OPCS strategy for conversion of its social survey work to computer assisted methods, and then to examine some of the second-level issues that have arisen in the early stages of implementing this strategy. The issues chosen arise in parts of the survey system where Blaise is the software used at OPCS. The issues are in the fields of authoring, testing instruments, documentation and computer assisted coding.

The paper refers in particular to the Employment Department's Labour Force Survey, the first survey for which OPCS used computer assisted methods of data collection. A companion paper by Jil Matheson discusses the more complex surveys which are now being implemented in CASM, using Blaise.

2. Computer Assisted Survey Methodology (CASM)

For some years the idea of computer assisted survey methods has been subsumed in the term computer assisted interviewing (CAI), covering telephone (CATI) and face-to-face (CAPI) modes, and interactive data entry (CADI or CADE). It was the possibility of cost-effective combination of CATI and CAPI that made computerisation of the interview feasible for agencies, such as OPCS, whose principal work would not allow them to rely on purely CATI techniques because of coverage bias and concerns over complex and lengthy interviews on the telephone. It is not surprising that the initial focus of discussion within and between agencies should have been on the obvious and dramatic effects of CAI on the data collection part of the survey process. Increasingly, however, attention is turning to the total survey systems of which CAI is only part[1].

OPCS's efforts and planning in this direction now go under the title of computer assisted survey methodology (CASM). The purpose of the reconceptualisation implied in the change of name from the earlier "CAI" is to ensure that all survey processes from design to dissemination are covered, and that all developments in the agency are considered in the light of computerisation of the central survey process of data collection.

3. Blaise and the QLFS system for data collection and processing

Computerisation of the interview became a possibility for OPCS when it discovered a package, Blaise, which provided the essential functions in a form that fitted the intuitive methods of survey designers, field managers and interviewers (and therefore was able to meet the cost and timetable criteria for production work). Computerisation of the interview did not have to mean treating its design as an extension of methodologies which are appropriate in the formal world of large-scale data management but which are too cumbersome for the gritty, nuanced, interactive processes of most social survey work. Confidence in the end product could be based on testing it rather than the means by which it was produced. This section explains how Blaise is used for the QLFS.

---

1) American Statistical Association 1991 Joint Statistical Meetings, Special Contributed Papers on Computer Assisted Survey Information Collection: International Progress - papers by authors from Netherlands Central Bureau of Statistics, Statistics Canada, OPCS Great Britain and US Bureau of the Census.

CAI at OPCS was initially developed and implemented for the former QLFS which converted from pencil-and-paper interviewing (PAPI) to CAI in September 1990. The former QLFS was replaced in March 1992 by a much larger QLFS of similar design.

The larger QLFS, like the former version, is a panel survey with sample rotation and a weekly placing pattern, covering every week of the year proportionately. Information is collected about all members of sampled households. Proxy interviews account for about one-third of all responses for adults. Interviews average about 30 minutes per household. The set sample yields over 60,000 responding households per quarter (about 160,000 persons per quarter). All first interviews are face-to-face and recall interviews are by telephone if the respondent agrees. Overall, nearly 60% of interviews are conducted through CATI (all from a central unit), and some 40% through CAPI (all face-to-face).

Response rates did not change with the introduction of CAI to the former QLFS. As in the PAPI era of the survey, about 83% of eligible households responded for first interviews and about 78% (of the original eligible sample) for fifth interviews. Item response was improved by CAI, as one might expect. Distributions were consistent with earlier quarters' results. Findings are similar for the enlarged QLFS.

The QLFS system is as follows, for every week of the year. Addresses are selected from a computerised file of postal delivery points (Postcode Address File) for small-volume users. The central case management system prints address lists and labels, and one-page summaries of each household to help interviewers plan their work; and it produces Blaise files with the corresponding serial numbers (and, for recall interviews, last time's data). These files are sent to the CATI unit or encrypted and dispatched to the CAPI interviewers. After the day's interviewing and completion of the post-interview work, such as coding occupation and industry, the data are transmitted to the central office. The laptop computers each contain quasi-case management, using facilities provided in a Blaise module for converting data to ASCII format. This ensures that only interviews which are complete, clean and have not been converted before are transmitted. The data are encrypted and compressed before transmission. The interviewers initiate these processes. At the end of the week they send in their encrypted interviewing and backup disks, which are searched for any missing interviews reported by the central case management system. Field managers decide which interviews to reissue for follow-up over the 9 days following the reference week. Data for completed weeks are passed on for updating (for the next interviews), derived variable creation and weighting to population estimates. The final data files must be passed to the customer, the Employment Department, within 4 weeks of receipt of the final interviews for a quarter.

The system outlined above could clearly be improved by full case management on the interviewers' laptop computers. In the present system, the interviewers cannot readily see which interviews have been accepted for transmission. Any discrepancies between what interviewers think they have sent and what has actually gone are only apparent at the end of the interviewing week when the central case management system still shows some cases as outstanding. Although the number of discrepancies is small, searching the relevant disks and resolving the problems takes a disproportionate amount of time. OPCS is interested in the new Blaise system, LIPS-SPIL, which involves 2-way communication and may therefore allow true case management for laptops.

4. Blaise and Authoring: Questionnaire and Edit (QE) instruments

One of the reasons for OPCS's choice of Blaise was its ability to deal with QE instrument changes rapidly at the level of the whole system, not merely of the field instrument itself. Definition of the field instrument in the Blaise language can be used to provide automatic definitions of the data wherever else they are needed in the system. This removes the need for reformat programs, which are notoriously error-prone and time-consuming. The idea of a single point of definition is built into Blaise, and OPCS computing specialists generalised it for packages to which Blaise does not provide automatic interfaces. In doing so, they abandoned traditional concerns of computing specialists for storing and processing data in the most machine-efficient ways. All LFS processing is on microcomputers, for which costs of "inefficient" database structures are far less than the costs for the time of skilled staff to program and test reformats.

The design for the enlarged LFS includes quarterly change of some 10% of the content. There is a core questionnaire which may be changed annually (with provision for emergency changes to reflect new legislation). The non-core questions, which vary quarterly, can be interwoven in their logical positions with core questions in a CAI QE instrument, rather than forming a distinct supplement to be completed after the core interview, as often occurs in PAPI surveys for logistical reasons.

Thorough customer testing is a vital part of CASM. With the CAPI software currently available it remains a distressingly labour-intensive process. OPCS is trying to develop ways to automate the process, with limited success so far. In the short term, however, the LFS needs a systematic and thorough approach to customer testing, which may fall short of the desired goal of automation. Currently there are 5 checks.

The first check involves a detailed (character by character) comparison of the new QE program in Blaise with the program from the previous quarter. Although labour intensive, this is a very useful exercise and can detect simple errors at an early stage of the questionnaire development process.

The next stage involves detailed comparison of the documentation provided by the Employment Department in the form of a QE specification and the questionnaire program in Blaise. The third stage consists of interactive checks of the functioning of the QE instrument on the laptop. The appropriate layout and content of questions are systematically checked, as are the appropriate use of checks and signals and the correct operation of all the standard functions used by interviewers in the field. Fourthly, the operation of hidden and protected fields needs to be routinely checked at this stage, which involves compiling a version of the QE instrument with all hidden and protected commands removed. Finally, it is vital to see the effect of any questionnaire changes on recall interviews, using a test batch of data brought forward in readiness for recall.

Although this system of customer testing has been developed specifically for the LFS, it is likely that the same principles will be adopted by other surveys moving to CASM in the future, until an automated alternative can be found. This is clearly a key area for development and one which OPCS considers a high priority as part of its strategy to move more of its surveys to CASM.

5. Blaise and Authoring: skills and new staff

OPCS aims to have the survey researchers who design, manage and analyse its surveys as the authors of CAI QE instruments, only resorting to specialist programmers in situations where the software is pushed to its limits and computing efficiency matters. Such situations can be expected to become increasingly rare as software improves. The argument for this strategy is that designing a QE instrument, in Blaise at least, involves the survey researcher in precisely the same essential steps as for a PAPI operation. The survey researcher will always have to specify what is wanted in some kind of formal language. The importance attached in most agencies to standards for paper questionnaires and edit specifications illustrates the need to squeeze out ambiguity and aid comprehension. Our experience is that, for surveys which are not pushing back the frontiers of CAI, writing Blaise instruments involves about the same level of knowledge of logic, special conventions and generalisable know-how as the paper questionnaires and edit specifications that we take for granted shortly after encountering them when we begin survey work. In these circumstances, for the survey researcher to write specifications for a programmer is a step backwards - reintroducing the possibility of error through miscommunication and, at best, duplicating effort.

The LFS QE instruments in Blaise have always been written and amended by survey researchers. The methods by which new recruits learn to write CAI instruments are much the same as for paper questionnaires and edit specifications. That is to say, training focuses on research concepts and their operationalisation: in relation to, such essential and difficult concerns, training in how to write in Blaise requires little time and is mainly a matter of learning the local conventions by understanding model instruments and reading the manuals, and practice.

The main problems for new researchers on the LFS are associated with the demands of the panel element of the survey. Sampled households are contacted five times at 13 week intervals. At each recall interview virtually all the data is carried forward from the previous interview and appears, as appropriate, on the lap-top screen. This allows interviewers to check that certain information given at the last interview is still applicable. Where no change in situation has occurred at a particular question the interviewer simply confirms the previous data entry. If a change has taken place since the last interview the new information is entered, overwriting the data brought forward.

The requirements of the system for recall interviews means that writing the LFS QE instrument is not simply a case of designing clear and concise questions, logical routing and sensible edit specifications with comprehensible error messages for interviewers. If the LFS was a straightforward survey without recalls this would be the case. Given the panel design, it is essential that the researcher has a very clear and detailed knowledge of its operation and a complete understanding of which elements in the Blaise QE instrument affect the structure and appearance of the questionnaire at recall waves. All editing for the LFS is done during the interview, so the Blaise QE instrument must be written to take account of the state of the data at the last interview, in the current one and at the start of the next. It is particularly crucial to ensure that data which must be preserved for the next interview never disappear from view as a result of new routing, lest the interviewer write off the interview before it can be retrieved. Dealing with such complexities means that the author of the QE instrument must have a very clear picture of the survey's structure. But Blaise itself presents no problems.

Thus new researchers have much greater difficulty with the conceptual problems of a panel survey which uses dependent interviewing and correction by overwriting than with learning to use Blaise. As noted earlier, training effort concentrates on survey design; Blaise needs and gets no special attention.

The skills of information technology specialists have been employed on the LFS in designing and implementing effective systems for backing up, storing, transporting, monitoring and ensuring the security and integrity of the outputs from Blaise. They have been particularly skillful and imaginative in building on the strengths of Blaise rather than attempting to fit its outputs into traditional models of data management.

6. Documentation and discussions with customers

The major redesign of the LFS questionnaire and edit instrument for the enlarged survey required extensive consultations between OPCS and the customer, the Employment Department. The Employment Department had to consult its own wide range of customers in other divisions and other government departments, most of whom had proposals for new questions and amendments to old ones. Draft instruments were vital documents in these discussions, but there were no paper questionnaires to fulfill this role.

The project manager at the working level in the Employment Department felt able to understand and work with the Blaise specifications for the QE instrument, after explanation of a few basic principles. The solution for the wide consultations was to use the printed questionnaire generated by Blaise, with some additions. This document lacks routing instructions, so information was added at each question about the subsamples to whom it applied. As Blaise works from precisely this information, and not from the programming equivalent of skip patterns, checking the discussion document against the Blaise instrument was less error-prone than checking complex skip patterns against customers' specifications tends to be with paper questionnaires. It may also be argued that this method provides analysts directly with the information they need about questions, and is preferable to requiring them to construct it by retracing skip patterns as they may have to do where paper questionnaires are used as documentation. We envisage that this form of documentation will be refined through practice to make it as readable as possible for a wide variety of audiences.

Blaise gives the survey researcher - and the customer, if that is someone else - much closer control over data quality than in systems (paper or otherwise) where questionnaires may be public documents but the equally important editing instructions are, if public at all, in languages which tend to be difficult for non-programmers to follow. The exact relationship of questions and edits (e.g. the order in which edits are performed) can be difficult to discern. In a CAI instrument, designers must consider fully the implications of edits as they design questions. Editing instructions must be comprehensible to the interviewers, who have to take action if they are triggered. Simple text must be supplied. The result is self-documentation of both questionnaire and edit in an accessible form which also shows the relationship between the two elements. Such opportunities that CAI offers for survey researchers will be lost if authoring is regarded primarily as a matter for good programming rather than for good survey design.

7. Computer assisted coding (CAC) in the interview

In the QLFS, the interviewer codes occupation and industry at home. When the QLFS started, nationality, country of birth and ethnicity also had to be coded; but the coding of these items was quickly brought into the interview with CAC, using the integrated Blaise module. The lists involved in CAC were short, with no more than 700 entries, and there was no effect on the speed of the interview. The enlarged survey has added the requirement for interviewers to code local authority district and travel-to-work-area of place of work for main job and job one year ago (address of firm is not collected in Britain); and subjects of educational and business qualifications. We expect to extend CAC in the interview to all the questions mentioned above. Trials have shown that the new Blaise CAC module can handle very long lists (more than 30,000 items for placenames, and similar sizes for occupation and industry) compactly and fast enough not to lengthen the interview.

8. Conclusion

Blaise has proved flexible enough, and easy enough to use, for survey researchers who are not skilled in programming to deal with the complexities of a panel survey. There are other ways of achieving our objectives than the ones we chose: for example, external files might have been used. In achieving our objectives for the QLFS, the close understanding which the survey designers have of their own requirements has been much more important to a successful survey using CASM than programming skills. However, there is a vital role for specialist programming support. It is to provide an environment in which the survey designer can have complete control over the details of the total survey system through QE specifications in Blaise. This takes full advantage of the central Blaise ideas on integration.

GENERATING MULTIPLE VERSIONS OF QUESTIONNAIRES

Mark Pierzchala
National Agricultural Statistics Service, Fairfax, USA

1. Challenges of versions of questionnaires

Any system that is to be considered for data collection and editing in the National Agricultural Statistics Service (NASS) must be able to handle versions of questionnaires. NASS is decentralized with headquarters in Washington DC and 45 state offices where most of the production work is conducted. For some surveys, there are potentially 45 versions of questionnaires that must be accommodated. The idea of generating versions of interactive editing and interviewing instruments by computer came from Wouter Keller of the Netherlands Central Bureau of Statistics. This paper presents the results of preliminary work done in NASS. It has been shown that a computer program can generate versions of instruments easily. Computer generated instruments have not yet been used in production.

2. Motivation for automated generation of versions

The Computer Assisted Survey Section in NASS has just been given a mandate to produce interactive processing instruments for editing, data entry, and data collection for all of NASS's surveys. This mandate is much expanded from just a few months ago when the section was called the CATI section. The CATI section concentrated just on Computer Assisted Telephone Interviewing for a subset of NASS surveys. With the expanded mandate, and with only a small increase in personnel in that section, NASS must re-evaluate its instrument production techniques. While it is possible to manually program instruments for versions of surveys the cost is very high. The cost must be measured in terms of both direct costs and opportunity costs. Direct costs involve time needed to program and maintain the versions of instruments.

Opportunity costs may overwhelm direct costs. These involve the implementation that is not carried out or is carried out on a delayed schedule because production and maintenance of the first instruments is so time consuming. By using an integrated system such as Blaise it is possible to reduce costs by programming one instrument for editing and interviewing. This represents a savings of about 50% over programming one system for editing and one for interviewing. This improvement is not good enough. It is also important to keep in mind the huge United States deficit which is about 400 billion dollars. Increased financial pressure upon NASS and other agencies is a certainty. The broader challenge will be to produce the same (or better) product in the future with fewer resources. By aggressive and intelligent use of technology this challenge can be met.

3. Example of versions of questionnaires

The Quarterly Agricultural Survey (QAS) best illustrates the challenge of multiple versions in NASS. Within each QAS questionnaire are several standard sections such as crops, grain stocks, hogs, and chickens. The crops sections of the December QAS versions offer the most extreme example of the challenge. The crops sections in the December quarter of the QAS will survey different crops in the various states as demonstrated by the three fictitious and simplified lists below:

STATE A	STATE B	STATE C
Corn	Irrigated Corn	Corn
Soybeans	Non-Irrigated Corn	Soybeans
Potatoes	Irrigated Sorghum	Alfalfa Hay
Sorghum	Non-Irrigated Sorghum	Sunflowers
	Peanuts	Tobacco
	Potatoes	
	Rice	
	Irrigated Soybeans	
	Non-Irrigated Soybeans	
	Cotton	

Several aspects of the version challenge are apparent:
1. there are different crops in different states,
2. there are different approaches to surveying a crop, especially in some states it is necessary to distinguish between irrigated and non-irrigated crops while in other states no distinction is made, and
3. the order of the crops differs from state to state.

Other aspects of the version challenge that are not as apparent include differences in question text, units of production (e.g., pounds, tons, or bushels), SAS variable names, and unique item codes that have to be associated with each question.

Various items of information are gathered about each kind of crop. For example for corn the items for each state above are:

CORN, STATES A & C	CORN, STATE B
acres planted,	irrigated acres planted
acres harvested for grain,	irrigated acres harvested for grain
production of grain,	irrigated production of grain
acres harvested for silage,	irrigated acres harvested for silage
production of silage,	irrigated production of silage
other corn acres.	irrigated corn acres for other uses
	non-irrigated acres planted
	non-irrigated acres harvested for grain
	non-irrigated production of grain
	non-irrigated acres harvested for silage
	non-irrigated production of silage
	non-irrigated corn acres for other uses

4. Building upon similar structures

There are similar structures between and within crops that can be used to simplify the task of generating versions. For example, the following structure, as exemplified by the following ROUTE statement, can be applied both to corn and to sorghum as well as other crops.

```
ROUTE    {consolidation of route statements from three blocks}

  Planted;                    {irrigated, non-irrigated, or both}
  If Planted > 0 then
    Harvestd                  {nested grain block}
    If Harvestd > 0 then
      Prodction               {nested grain block}
    endif;
    Harvestd                  {nested silage block}
    If Harvestd > 0 then
      Prodction               {nested silage block}
    endif;
  endif; {planted}
  OtherUse;
```

A simpler structure that can be applied to several crops such as soybeans, potatoes, and tobacco is:

```
ROUTE                         {all one block}
  Planted
  If Planted > 0 then
    Harvestd                  {one use of crop only}
    If Harvestd > 0 then
      Prodction
    endif;
  endif;
```

Edits in the CHECK and SIGNAL paragraphs have similar form between crops and states, even if details differ. For example:

```
SIGNAL
((softlow < Prodction/Harvestd) and (Prodction/Harvestd < softhigh))
```

Notice that no matter what the crop (and sometimes even twice in the same crop), the same questions names are used. This is deliberate as this will make excellent use of blocks (and nested blocks) in Blaise and also of its "dot notation" and significantly cut down on the coding and maintenance of the instruments. Also note that the SIGNAL paragraph has edit limits that are defined with variables, not hard-coded numbers. This allows dynamic definition of edit limits between crops. The idea is to build upon similarities between versions of questionnaires, not to agonize over differences.

5. Breaking down the version challenge into manageable parts

The strategy used in generating versions of instruments is to separate aspects of programming that change from crop to crop from those that do not change from crop to crop. In Blaise this separation is made very neatly. The things that do change from crop to crop appear mostly in the QUEST paragraph, while the things that stay the same appear totally in the ROUTE, SIGNAL, and CHECK paragraphs. Thus the problem is simplified if a way can be found to apply the same ROUTE, SIGNAL, and CHECK paragraphs to different QUEST paragraphs. Conceptually this may be done in one of two ways:

1. create a generic QUEST paragraph that is dynamically redefined for each crop, or
2. create a different text file for each crop's QUEST paragraph.

The first option is mostly but not totally possible in Blaise. It would require many expensive text substitutions and an external file that would hold the text. Also Blaise does not allow some parameters to be dynamically defined such as question numbers (item codes in NASS) and question labels. For these reasons and others, this demonstration has been constructed using the second method. Examples of QUEST paragraphs are as follows:

```
QUEST  {for corn}
  Planted  "/100/  How many acres of SOYBEANS were planted for all
   purposes?" "CSYB__PL" : 0..997;
  Harvestd "/101/  How many acres of the $PLANTED acres for SOYBEANS
   were harvested?" "CSYB__HV" : 0..997;
  Prdction "/102/  What was the production of SOYBEANS in BUSHELS?"
   "CSYBPROD" : 0..9999997;

QUEST  {for tobacco}
  Planted  "/200/  How many acres of TOBACCO were planted?"
  "CTOB__PL" : 0..99.7;
  Harvestd "/201/ How many acres of the $PLANTED acres for TOBACCO
   were harvested?" "CTOB__HV" : 0..99.7;
  Prdction "/202/ What was the production of TOBACCO in POUNDS?"
   "CTOBPROD" : 0..9999997;
```

About the only thing that is the same between the two QUEST paragraphs are the question names. The item codes, the SAS variable names, the units of production, the valid values, and even some of the question text differ. However, the same ROUTE, SIGNAL, and CHECK paragraphs can be applied to both QUEST paragraphs. This is done by use of blocks in Blaise and by use of INCLUDE statements as illustrated below:

```
BLOCK crn;
  QUEST {for corn}
  INCLUDE "RSC1.fle";  {holds ROUTE, SIGNAL, and CHECK paragraphs}
ENDBLOCK;
QUEST  corn : crn;

BLOCK tob;
  QUEST {for tobacco}
  INCLUDE "RSC1.fle";  {holds ROUTE, SIGNAL, and CHECK paragraphs}
ENDBLOCK;
QUEST  tobacco : tob;
```

The INCLUDE statement is used here to repeat the same code twice. The reason that Blaise does not confuse questions between the two blocks is because of the dot notation that Blaise uses. For example the names of the planted acres questions defined above are CORN.PLANTED and TOBACCO.PLANTED.

6. Customization of blocks

It remains to customize each block as regards edit limits (and perhaps a few other things as well). This customization is done in the ROUTE paragraph at the instrument level (highest level of organization) as follows:

```
ROUTE                                    {instrument or questionnaire level}
  ID;                                                {Identification block}
  Land;                                              {some land questions}

                                               {compute edit limits for corn}

    Compute softlow := 50; Compute softhigh := 150;
    Compute hardlow := 0;  Compute hardhigh := 300;
  Corn;                                              {call corn questions}

      {compute edit limits for tobacco}
    Compute softlow := 150; Compute softhigh := 450;
    Compute hardlow := 0;  Compute hardhigh := 800;
  Tobacco;                                        {call tobacco questions}
ENDQUEST.
```

In practice, the variable edit limits are calculated from an external file as they change from state to state for the same crop. The code above is a simplified extract of what one version of an instrument would look like. It can be programmed by people but there is no reason that major parts of it cannot be automatically generated. Remember that there are potentially 45 versions of the questionnaire to produce. Manual coding would be tedious, time-consuming, and subject to error. Even though this method is better than current methods used with another system in NASS, it can still be improved.

7. Specialization of tasks

The nice thing about the automated approach is that in separating the QUEST paragraphs from the ROUTE, SIGNAL, and CHECK paragraphs, different people can build and maintain them. For example, the QUEST files can be the responsibility of secretarial and clerical staff because they are simply text files of set and structured format. The ROUTE, SIGNAL, and CHECK paragraphs can be built and maintained by subject matter specialists.

8. Requirements for automatic generation of versions

Three major requirements for automatically generating versions of instruments are a library of code, a parameter file of specifications, and a program that generates the versions of instruments. The library serves as a resource of coded segments that the generator program assembles according to the parameter file of specifications.

9. Library of code

The concept of a library is very important. By breaking up the Blaise coded segments as described there are potentially hundreds of small files where before there were just several larger ones. They cannot all be placed in one directory. They must be placed in a simple subdirectory tree whose structure is readily readable and understandable to those who must work with it. Also, the file names must conform to a naming convention which is also understandable. The directory tree which makes up the library is displayed:

```
BLIB    CROP1                     {1st quarter crops}
        CROP2                     {2nd quarter crops}
        CROP3                     {3rd quarter crops}
        CROP4     CORN            {4th quarter corn}
                  SORGHUM         {4th quarter sorghum}
                  etc.

        R_S_C                     {Route, Signal, Check paragraphs}
        HOGS
        STOCKS
        etc.
```

BLIB stands for Blaise LIBrary and R_S_C for Route, Signal, and Check. Within a subdirectory such as CORN there are potentially several files of QUEST paragraphs, one for each way that corn may be surveyed across the country in the December quarter. File names must describe (even if cryptically) what they contain. The first part of the two part file name describes the crop and the approach to surveying it. The extension of the file name will be used to designate a file that contains minor modifications to one of the general survey approaches if required in a particular state. For example:

```
CORN1G
CORN1S
CORN1G.55
```

CORN1G holds a QUEST paragraph concerning approach 1 of surveying corn for just the Grain questions. CORN1S contains similar questions for Silage. These two files are in fact treated as the default (or master) files for corn that would normally be used in all appropriate states. However if state 55 (Wisconsin) needs different wording or range of valid values (but not different question names, item codes, or SAS variable names) then CORN1G is copied to CORN1G.55 and the latter file is modified. The version generating program would then choose CORN1G.55 over CORN1G when generating the version for state 55.

For each structure that is found in the QUEST paragraphs there must exist a corresponding R_S_C file that sets the ROUTE, SIGNAL, and CHECK paragraphs. For example, files CORN1G and SORG1G (corn and sorghum respectively) have the same structure (but different details). A file found in the R_S_C directory that would apply to both CORN1G and SORG1G would have a descriptive name such as ROW1G.FLE which would stand for "ROW crop 1 for Grain". The extension here has no special meaning but serves as a convenient way for searching for certain kinds of files in the library. While there are many QUEST files for crops there are only several R_S_C files because relatively few structures are needed to cover all the crops.

10. Parameter file of specifications

The parameter file(s) of specifications must contain pertinent information for each state. Ideally there would be one file where each state's specification would be contained on one line. The specification file can be either an ASCII file or a Blaise file. The former might be generated from the agency's specification system, the latter from a subsidiary

Blaise instrument especially designed for hand entry of specifications. NASS's current file of specifications (used for SAS editing and summary) falls a little short of what is needed for the automatic generation of Blaise instruments. Each state's specification must contain a list of valid item codes (which is equivalent to stating which crops are valid for that state), SAS variable names, the order in which crops are to appear, and the state's ID number. It may be necessary to have most of the information in an ASCII file and other information such as order of crops designated in a subsidiary Blaise instrument. If a subsidiary Blaise specification instrument is necessary then it would be designed so that a clerk could fill in the proper information.

11. Generator program

The program that generates the versions from the library of code and the parameter file need not be very complicated. It needs to be able to read information from the parameter file and draw upon the code from the library. The generator program can be written either in Manipula (a Blaise file manipulation utility) or in a programming language such as Turbo Pascal. All work so far has been done with Manipula. It is able to read either ASCII or Blaise files directly and to write text files (including DOS BAT files) based on the information in the parameter file. The sequence of events is as follows:

- The generator program reads a line from the parameter file.
- Lines are written into a text file by the generator program. The lines in the text file refer to code from the library.
- The Blaise program is built up from a standard front part, the text file of library references, and a standard ending part.
- The Blaise program is syntax checked and compiled from DOS.
- The compiled program is copied into the proper holding directory.

The lines that are written into the text file are mostly INCLUDE and QUEST statements. A portion of such a program is as follows:

```
INCLUDE '\blib\crop4\corn\corn1g';  {corn for grain questions}
INCLUDE '\blib\R_S_C\row1a.fle';    {route and edits}
QUEST   grain : firstuse;           {end of first nested block}
INCLUDE '\blib\crop4\corn\corn1s';  {corn for silage questions}
INCLUDE '\blib\R_S_C\row1a.fle';    {route and edits}
QUEST   silage : secnduse;          {end of second nested block}
INCLUDE '\blib\R_S_C\row1.fle';     {end of whole block}
```

These lines represent the code from the library that will ask about 10 questions, route them properly, and apply approximately 10 edits.

The person who writes the generator program does not have to be a systems person but should have some facility with computers. The generator program must be written in a very robust way. It should have capability to add new crops and new sections of the questionnaire without alteration. In fact, once written, it should apply to all quarters or periods of a particular survey and run for several years without maintenance. NASS does not want to trade a problem of maintaining instruments to one of maintaining a generator program.

The generation of versions of instruments is done in batch, one version after another. Best results are obtained with a powerful computer (486 processor or above) with enough RAM and extended memory to hold the library of code, the Blaise system files, Turbo Pascal, and work space for the syntax check and compilation. Probably 16 Mb would be sufficient. Time needed to generate each version is approximately 1 minute. Computers with this configuration are now starting to appear on desks in NASS. The only time the hard disk is used is in storing the compiled instruments.

If problems are found after the versions of instruments are generated, then repairs are not made to the instruments, but to the library of code or to the generator program. After repairs are made there, appropriate versions are regenerated.

12. Survey management

Survey management is standardized across state offices and across versions by use of standard front part of the instruments that includes a block of code for management purposes. This management block contains survey control questions, many of them with the selector attribute. Management tables in Abacus or Manipula are applied across versions of instruments and even across different surveys. Each state produces the same tables and accesses or moves forms (with the Blaise forms manager) with the same selector questions. The only difference between the versions of the instruments is the subject matter.

13. Instrument administration

There has been some question concerning the administration of many different versions of instruments from headquarters. This need not be more difficult than current practice and should even be easier given proper management. The idea is to build the instruments correctly in the first place and avoid having to repair them in mid-survey. Some survey administration could be automated. For example, a computer program could distribute instruments to the correct location. A database of information about the instruments already exists in the parameter file of specifications. This information can also be used to help administer the survey. For example, if there is a problem with a block of code that concerns only 4 states, then the database could be used to designate which 4 states are concerned. Repairs are posted only to the 4 states, the other 41 states are not bothered. In the current system if there is a repair to be made then all states must update their instruments, even if it does not concern their survey program.

Testing different versions of an instrument is an issue. The question is how much testing is needed. The key is not to wait until versions of instruments are automatically generated before testing. The time to test is before chunks of Blaise code are put in the library. The library should consist only of tested code, so when these chunks are pulled into an instrument they should be regarded as tested subroutines. However, there are still things that might go wrong. For example, some of the crops may not have been placed in a version of the instrument, or a few edits that go across crops may not work. The Blaise setup generator will help with some of this testing. It can produce reports of item codes and SAS variable names (with NASS customized setups) for each version of the instrument. These can be manually inspected, however a computer comparison of the reports against the original specification file would be preferred. The dictionary provided by Blaise as well as the technical description will be very useful as well. The testing that remains should be carried out according to a strict protocol and may be done by clerical personnel. Inevitably the final testers will be the users in the state offices as is done now. One point to keep in mind is that if instruments are generated automatically then there is more time to test them afterwards but less need to do so.

14. Multiple versions versus one massive program

Some people wonder why multiple versions of an instrument should be generated. Why not produce one massive instrument that can handle all situations and use the parameter file of specifications to determine which questions to ask in which state? In fact this can be done. The computer generation technique need not produce many versions of one instrument. The approach can be used to produce one instrument that invokes the right questions in the right state. However there are two strong reasons for generating multiple versions. First is that Blaise is a forms based system with two modes of operation. The interviewer should not have to see many or any inappropriate questions in the forms (bottom) part of the screen even if they are not invoked. It is possible to use the NEWPAGE question in order to keep inappropriate questions from enumerator sight, but this makes less efficient use of the form, adding pages unnecessarily. The problem becomes worse in the editing (CADI) mode where the routing is passively enforced, not dynamically controlled. The data editor would have to deal with up to three times as many pages, many of them unfilled, as would be necessary in a customized instrument.

The second reason that individual versions are preferred relates to unusual aspects of NASS's survey population, United States farms. The sampling universe is relatively small and getting smaller, and farm operations can be very large and diversified. As a result some farms are surveyed several times during the year for one survey or another. This situation worsens every year. As several surveys are conducted at any one time it is desirable to contact the farmer one time (within some period of time) and to ask all appropriate questions for all appropriate surveys in one session. Currently this means that the first survey is administered in CATI but then the interviewer leaves CATI and administers the rest of the questions on paper. For the second and subsequent surveys the benefits of CATI are lost for that farmer and the data must then be entered into the system causing more work. A solution to the multi-survey problem is to construct instruments that can handle multiple surveys. By customizing versions for each state, the size of each instrument is kept manageably small allowing more modules to be added for other surveys. A parameter file is used to control the interview and invoke the correct surveys in the instrument. This method should reduce respondent burden by asking certain questions only once. For example, if the name and address information is verified for one survey in the interview, then it will not be necessary to verify it for other surveys in the same interview. The handling of the data after data collection would be facilitated by the use of subfiles available in Blaise. Each survey's data will be kept in its own set of subfiles. A powerful feature of Blaise is that survey management can be carried out at the subfile level in addition to the management at the form level. The generation of MEGA-VERSIONS of interviewing instruments is a straight forward extension of generation of versions of instruments as covered in this paper. However there are many interesting ramifications as regards data handling and survey management. These will be the subject of a future paper.

15. Improvements in Blaise

There are a few enhancements to the Blaise system that would help in the generation of versions of instruments. These include the capability of passing question numbers and question labels as parameters, allowing question numbers to be used in tables, nesting of an INCLUDE within another INCLUDE (two levels only would be sufficient), and a new kind of question similar to NEWPAGE called NEWCOL. The NEWCOL question would help with formatting the screen between the CATI/CAPI mode and the CADI mode when the columnar format for questions is used. It would start the next question at the top of the column. Currently it is difficult to make the screen look as nice as possible in both modes. None of these enhancements is critical to the success of this method of generating instruments, they would just make life easier.

USING BLAISE FOR COMPLEX SURVEYS OF HOUSEHOLD FINANCES*)

Jil Matheson
Office of Population Censuses and Surveys (OPCS), London, England

1. Introduction

Following the successful conversion of the Labour Force Survey to computer assisted methods, attention has turned to extending these techniques to other OPCS surveys. This paper briefly describes developments to date on two particular surveys - the long-standing Family Expenditure Survey (FES) and the new Family Resources Survey (FRS). There is considerable overlap between the two surveys, each requiring detailed information about household and personal finances. The complexity of these surveys has presented new challenges for computer assisted survey methods (CASM) and these are described.

A companion paper by Tony Manners and Nikki Bennett discusses OPCS's strategy for implementing CASM more generally.

2. Family Expenditure Survey

The FES has been running continuously since 1957 and is an extremely important source of information for the British government. It is carried out by OPCS on behalf of the Central Statistical Office with the main aim of providing expenditure weights for the Index of Retail Prices. It is also widely used for other purposes such as modelling the effects of tax and benefit changes, assessing take-up of state benefits, and estimating consumer expenditure. It comprises an interview lasting

*) Based on a paper published in Survey Methodology Bulletin, OPCS, July 1992 : Further developments in Computer Assisted Personal Interviewing for household income surveys. Jean Martin & Jil Matheson

between 1 and 3 hours which covers income and major items of expenditure, and an expenditure diary kept by all adults for 2 weeks. All adults in the household must co-operate fully for the household to be counted as responding. The response achieved is generally around 70%, some 7000 households a year cooperating.

The interview starts with a household questionnaire which is completed with the householder and his/her spouse. This questionnaire establishes basic details about the members of the household and covers regular household expenditure on housing, gas and electricity, household insurance and other regular expenditure, and also large occasional items of expenditure for which 2 week diaries cannot provide reliable estimates.

Individual interviews are then completed for each adult. These cover all items of income - earnings, benefits, pensions, investments etc.

Great emphasis is placed on obtaining as complete and as accurate information as possible. Respondents are encouraged to consult documents such as pay slips, rent books, bank statements, bills etc.

The main pressure on the FES which led to consideration of changing to computer assisted personal interviewing (CAPI) was the requirement for results to be available faster and to obtain better quality data without unacceptable extra costs. Currently clean data from the FES are not available until about 20-25 weeks after the end of each quarter.

Collecting detailed and accurate information about people's financial affairs is demanding for both interviewers and respondents. Data collected currently on the FES suffer from missing information, errors and inconsistencies which necessitate a very extensive and time consuming post-interview editing system. It is hoped that CAPI will remove many of these errors. For example, over half of all failures of edit checks result from routing errors leading to questions being missed; with CAPI these should be eliminated.

A further major source of error occurs when the sum of individual items and totals are inconsistent. Not only is it unreasonable to expect interviewers to carry out a lot of arithmetic in the interview but often the amounts for different items are for different time periods and need to be converted to a common base before they can be added. Interviewers cannot do this but such checks can easily be programmed in Blaise.

Despite the expected improvements CAPI can make in some areas there remain intractable problems. Surveys such as the FES are limited by the amount of information respondents are willing and able to provide. In many cases respondents do not have full details of their financial affairs, either because they have not kept the relevant documents or because they are kept by their accountant.

Work started last year to assess the feasibility of changing to CAPI for the FES interview and to consider how the expenditure diaries might be integrated. The questionnaires were programmed in Blaise and the first small CAPI field trial took place in March 1992. This trial aimed to highlight any major difficulties at an early stage, and point to the main features of the CAPI interview which would require further investigation and development. It was also felt to be important to get an early indication of public and interviewer acceptability, although the trial was not designed to test response rates.

Interviewer and respondent reaction to the trial was generally very positive and it was concluded that CAPI would be technically feasible for the FES although further work is needed before a final decision on whether to switch can be made. Various areas for improvement were identified and these are discussed below.

3. Family Resources Survey

This new continuous survey, covering some 25000 households a year, was specified by the customer department (Department of Social Security) as a CAPI survey from the outset. Early development work had established the feasibility of this mode and recommended that Blaise be used. The main survey, to be carried out jointly by OPCS and Social and Community Planning Research (SCPR), begins in October 1992.

The survey covers many of the same topics as the interview part of the FES but its larger sample size will allow specific groups such as low income families to be studied in more detail, and the interview will cover more details of income but little on expenditure. A major requirement of the survey is that results are available quickly, hence the decision that the survey use CAPI. Edited data will be delivered to the customer within 13 weeks of the end of fieldwork, although some further work to impute missing values and calculate derived variables will then be necessary before analysis can start.

Because of the similarities between the FRS and FES in terms of both content and the problems they pose for CAPI, the development work is being co-ordinated as far as possible, bearing in mind that OPCS is solely responsible for the FES, while SCPR is responsible for the development of the FRS questionnaire instrument.

Using Blaise for the FES and FRS

The methods and procedures developed over the years for FES interviews, which will broadly be the same on the FRS, pose particular problems for CAPI. Our experience in pilot work is discussed below.

Concurrent interviewing

On the FES, interviewers attempt to talk to all adults together in order to persuade them all to complete the expenditure diaries; households are only counted as responding if all adults co-operate fully. Interviewers generally interview two, and sometimes three, people at the same time. Concurrent interviewing works by the interviewer reading out a question and recording the answers from both individuals. If their answers lead to different routing the interviewer may complete a short block with one person before returning to the other. This has the dual advantage of keeping to a minimum the time spent in the household and helping to get complete and consistent information. This is particularly important in the case of couples whose affairs are closely inter-related. The procedure minimises the risk of missing items of expenditure or income or of double counting, and allows the person with most knowledge of the financial affairs to provide the information. For these reasons it is also planned to use concurrent interviewing on the FRS.

A method for permitting concurrent interviewing was field tested on both surveys. This consisted of blocks of related questions in table format so that the interviewer records answers for both respondents before moving on to the next block. While generally successful, in practice interviewers recommended that blocks of questions should be made shorter. This will be tested in the next field trials.

The Blaise program for the individual questionnaires permits interviews with two adults, while the paper and pencil FES questionnaire currently allows up to three adults to be interviewed concurrently. Interviews in larger households have to be carried out consecutively.

For various reasons it is felt to be desirable to limit concurrent interviewing to CAPI to two individuals. This makes programming easier, and the response time of the program faster. But perhaps more importantly, discussions with interviewers revealed that, in practice, they often prefer to interview no more than two people at a time. This limits the amount of time each individual has to be present (although not the length of time the interviewer is in the household), makes the interview easier to control, and aids intra-household confidentiality. This procedure is being adopted by the FRS and is felt to be appropriate too for the FES.

4. Flexibility to move around

Another feature of the interviews for both these surveys is the need for interviewers to have flexibility to move around the questionnaire, particularly to go back and enter data which was not available earlier in the interview. The requirement that the data be as complete and as accurate as possible means that interviewers encourage respondents to consult documents wherever possible - bank statements, pay slips, rent books etc. But often it is easiest if the respondent goes to find all these documents in one go so the interviewer leaves questions blank until the information is assembled. This may even be on another visit.

The majority of interviewers felt that after some practice in the quiet of their own homes they could easily navigate their way round the questionnaires. An easier way of jumping straight to missing values is required however.

Another possibility is that one respondent leaves the room to find some documents while the interview continues with the other respondent. This is handled by inserting a question for interviewers between blocks asking them to indicate if both respondents are still present. If not only the questions applying to the respondent who is present appear. Again this was fairly successful in the field tests but the number of such interviewer questions has now been reduced to improve the speed of interview.

5. Complex checks

CAPI provides the opportunity to check answers in the interview when respondent and interviewer can sort out problems together. However, with such complex surveys there is a limit to how many checks it is feasible to introduce without overburdening the interview. Moreover, there are some inconsistencies which cannot be resolved; respondents are not always consistent in their answers. We aim, therefore, to include only those checks which can be resolved during the interview and to run other checks after the interview. One possibility is for interviewers to run some additional checks at home when they have time to think about the problem and to recontact respondents if necessary. Indeed, FES interviewers always call back at responding households at least twice after the initial interview to check diary keeping. A separate 'checking' program for interviewers to run at home between visits, including additional checks and identifying missing values, is being developed for the next FES trial. Part of the development process is to make decisions about the most appropriate place to carry out different checks on the data: in the interview, by interviewers at home, or in the office.

6. Length of interview

Both the FES and the FRS involve very long interviews so it would not be desirable for CAPI to increase the length. This means that the programs must run fast enough that interviewers are not waiting for the next screen to appear. Recent field trials have used a Toshiba T2000, which has a 286 processor and a 20Mb hard disk and runs under DOS 5.0. Interviewers have commented that the introduction of some complex arithmetic checks in recent trials seems to make the interview slow in parts.

7. Keying accuracy

Another concern about the introduction of CAPI was whether the interviewers would be accurate at keying in answers, particularly as so many involve entering amounts of money where it might be easy to enter an incorrect digit without noticing. As well as range checks, signals have been added for implausible answers.

8. Changes to the questionnaire

An important requirement for both surveys is that it should be easy to change the questionnaire instrument and introduce new questions. Both surveys expect major revisions every year and some revisions each quarter, mainly to cope with the effects of changes in benefit rates etc which affect checks if not actual questions.

One advantage we have found of using Blaise is that it is easy for research staff to learn and to write moderately complex questionnaire instruments, although we have found that for complex surveys, like the FES and FRS, small teams of researchers and programmers work best as their skills are complementary.

9. Interviewer notes and comments

Because of the complexity of the information being collected, on the paper survey FES interviewers normally make notes on the questionnaire, both for their own use and to help the coders and editors in the office. It is impossible for the questions to cope with all circumstances and so interviewers make notes when the questions do not allow for a particular situation and expect the office to decide how to handle it. This posed a problem because the facilities for making and retrieving notes were not ideal in the version of Blaise used in field trials so far. Interviewers could open a comment screen at any question and enter free text. This is stored in association with the question. The problem is that interviewers could not then reaccess the comments either to amend them or even to read them. So this facility was no use for notes the interviewer needed to refer to during the interview and she or he had to keep a notebook handy and make notes on paper. The comment facility was only used therefore for notes to the office coders. Even at the office we have found it easier to strip out the comments and print them on paper rather than expecting coders to read the comments on screen.

The new version of Blaise seems to have a much improved commenting facility but this has not yet been tried in the field by OPCS.

10. Conclusion

The two surveys described here are, in some ways, significantly more complex than most others carried out by OPCS. They place considerable demands on respondents and interviewers, yet analysts require high quality and timely results. This made them good candidates for early investigation of the feasibility of converting to CASM, although we did not at first know whether their special features would place too heavy a demand on Blaise. Further development work is planned for the FES, including looking at computer assisted coding for expenditure diaries, and a formal comparison of estimates produced by the two modes. We are hopeful that the FES will switch to CASM by 1995 at the latest. The FRS has already undergone a successful period of development and is about to begin mainstage fieldwork, using Blaise.

BLAISE IN A DANISH CONTEXT

Johannes A. Noordhoek and Leif Bochis
Danmarks Statistik

1. Introduction

Danmarks Statistik has based most of its data collection on the use of administrative registers. This means that the collection of data on demographic and social topics through interviews is limited to a few surveys which supplement register-based statistics. The following surveys are carried out:

1. *Labour force surveys*
   So far this survey is based on a sample of 15 000 households where all persons between the age of 15-74 years are interviewed. Telephone interviews are supplemented with mailed questionnaires to households not accessible by telephone. The survey is conducted every year. In 1992 and earlier years interviewing was done during the first half of the year, but in the future interviewing may be extended to cover the whole year, in connection with an enlargement of the sample to 60 000 persons.

2. *Omnibus or multi-purpose surveys*
   These surveys are carried out every month with a sample of 1 800 - 2 300 persons aged 16-74 years. As the name suggests these surveys cover a wide variety of topics. Users of the survey are typically government institutions, but the core of the survey is the Consumer Survey for the EEC. Starting in the fall of 1992 the Omnibus surveys will also contain questions about transportation patterns. All interviews are done by telephone.

3. *Family budget surveys*
   So far these surveys with a sample of 5 000 households have been conducted every 5-6 years. The latest family budget survey for 1987 was based on personal interviews supplemented with diary data on expenses. In the future, however, family budget surveys may be carried out more frequently, and in such a way that each year about 1500 households will be interviewed. The cumulated results are then published every 3 years.

4. *Ad-hoc surveys*
   Government institutions and other organizations may want to conduct a survey of special populations based on samples from the administrative registers. The surveys may be conducted by mail or by telephone interviewing.

The following considerations led to the adoption of computer assisted telephone interviewing (CATI) using a centralized team of interviewers.

CATI could reduce the total cost of data collection and data processing. Earlier we still used face-to-face interviews in part of the omnibus surveys as a supplement to telephone interviews. It was decided to drop this supplement which so far had made it necessary to have a decentralised interview organisation. We realised that the sole use of telephone interviewing would reduce the total response by 7-8%. This reduction would have to be compensated by weighting the results. The main cost reduction would, however, come from the integration of interviewing and data-editing in one process, leaving only a limited amount of subsequent data-processing such as coding of answers to open-ended questions.

CATI's integration of several processes in one would make it possible to make the results of the surveys faster available to users. In particular outside users made increasing demands for fast delivery of the results.

Finally, the establishment of a centralized interview organization would make it easier to control the quality of the data collection by increased monitoring of the interview process.

A review of some of the existing CATI-systems on the market indicated that we would be best served by a system used by a central statistical office. It would have to be a system which could handle relatively complex surveys in regard to the contents of the interview, as well as the administration or management of interviews with households and single persons. We finally settled for Blaise, even though it did not, at that time, have an efficient call management system. On the other hand, Blaise was relatively well documented and had also developed a program which could be used for entry of data from mail surveys.

After some experiments in 1990, we started using Blaise-CATI for telephone interviewing during the second half of 1991. For this purpose, we established a small interview organization with 16 interviewers, which number was later extended to 28.

The interviewers are working in 3 rooms with 8 PC's in each room under the supervision of a supervisor, who can assist with problems arising during the interview, and who also has facilities to monitor the interviews. All 24 PC's are connected to a LAN-Manager network.

Since the start we have completed about 10 000 Omnibus interviews and about 20 000 Labour force interviews. We have also used Blaise-CADI to enter data from a large mail survey with 2 500 completed questionnaires and data from a smaller business survey. Despite some technical difficulties, we have, in general, been quite satisfied using Blaise-CATI as well as Blaise-CADI. The technical difficulties were in part due to the fact that the introduction of Blaise coincided with the installation of a new network. This resulted occasionally in breakdowns and/or long response times. An upgrading of the LAN-Manager network in spring 1992 led to some 'locking' problems. This manifested itself in such a way that if more than 4 interviewers were working on the same batch of addresses, it would lead to long response times and finally breakdown the system. We have temporarily solved the problem by letting a maximum of 4 interviewers work on the same day batch. In the meantime, we are working on a more permanent solution with the co-operation of CBS.

As previously mentioned, our experiences with Blaise have been satisfactory. However, we do have some suggestions for improvements, which would make life easier for the staff worker who designs the questionnaire for the interviewers and for the end-users of the results.

Designing the questionnaire could be more user friendly. It would be convenient if the system could produce a flow-chart showing all the skip-patterns. Blaise does not use a record-based approach but treats the whole program as a closed system which is then compiled. This makes it more difficult or at least time consuming if a small detail in a question has to be changed. This is particularly important when interviewing has already started and minor but important changes are necessary. From the viewpoint of the interviewer it should be easier to avoid punching the wrong answer code. At present the interviewer is not alerted if a wrong answer code is within the valid range.

If the whole answer text were highlighted after selection and then entered by punching 'ENTER', it would focus the interviewers attention on the screen and therefore avoid punching the wrong answer code. In particular for questions where multiple answers are possible this would be a great improvement.

For the end-user of the data it would be convenient if a code book were produced which would clearly describe the questions, the values and descriptions of the answer categories as well as the routing to and from the question.

2. Experiences with the Call Management System in Blaise

At Danmarks Statistik we have used CATI with its Call Management System for the following two tasks.

1. Omnibus surveys, which are monthly surveys based on samples of 1 800 - 2 300 persons.
2. The Labour force survey, which is a household survey for which the data are collected over 13 weeks in the first half of the year. The sample is a random sample drawn from the central population register. After the sample is drawn, telephone numbers are found by a semi-automatic process which searches through the databases of the telephone companies. The sample consists of about 25 000 persons in about 15 000 households and telephone numbers are found for approximately 85% of the households.

3. The household interview system for the Labour force survey

Our job was to develop an interview system to collect data from household members in a quick way, meeting our requirements. We based our work on the following conditions.

1. We should use the Blaise Call Management System in order to be able to use the routines for appointments (including automatic allocation of telephone numbers), supervisor utilities to identify and treat forms by number and telephone number, and the reporting facilities about responses, etc.

2. Each interviewer should be able to use the available knowledge about the composition of the household to collect data from all eligible members of the family.

3. It should be possible to use data from the system to calculate interviewer remuneration. Each interviewer is paid a fixed price for each completed interview.

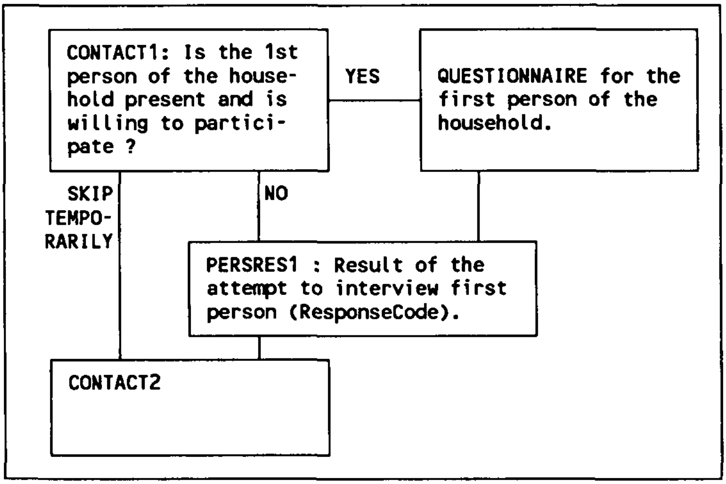
4. It should be possible to use the system in an easy and fast way for all interviewers and it should not lead to loss of data or other problems in its application.

According to these conditions we constructed the interview system as one logical questionnaire for each household. In this questionnaire the main questionnaire acts as a kind of household administration system and the actual questionnaires are entered as subquestionnaires for 1-8 household members using Blaise's subfile concept.

Telephone numbers and general household information was entered - according to the Blaise file structure - in the APPOINTMENT block and other information about household composition as well as background information was entered in an external reference file, which could guide the interviewer and check whether all relevant members of the household were interviewed.

When interviewers call a household, the most convenient procedure is to interview the person who answers the telephone first. In other words, it should be possible to shift quickly to the relevant person, complete the interview and then ask for one of the other family members. Since each member of the family has certain background information and should be able to be identified subsequently, we considered it best to maintain a structure where each family member has a fixed place in a rank order. This means that the first subquestionnaire refers to the first person in the family, the second subquestionnaire to the second member, etc. We chose to implement the household administration as a series of questions about each person's participation and response categories. Thus, with a single key press the interviewer could pass the first person in the family and interview the second person and then skip back to the first person (see figure 1).

Figure 1. Structure for the Labour Force Household Survey



Question CONTACT1 - whether the first person is present and wants to participate - can be answered with either YES, followed by the person, or NO, followed directly by the question on the relevant answer code or by the Skip Temporarily question, which can be used, if the interviewer first contacts person 2 or 3, etc. The text to the question CONTACT1 (and to CONTACT2, etc.) consists of a list of the household members, including response codes entered during an earlier contact with the family. We also supplied questions CONTACT1, CONTACT2, etc. with appropriate labels (IP1, IP2, IP3, etc.) which made it possible for the interviewer to skip from one respondent in the household to the next. This facility is especially useful for interviewers who contact a large family, that has been contacted before and where some interviews have already been done. In that case it is necessary to skip some of the subquestionnaires.

Before completing the household interview a check is made whether or not the response code 'Skip Temporarily' was removed. The last entered code is checked as well as whether an appointment for a later call has been made. This response gave some problems, because it was not possible to store the response to the first contact with the family. Therefore some interviewers used this facility to skip some family members when calling again, instead of skipping to a label. Since it is possible to make appointments during an interview, it was not possible to ensure that a check was made for this response code when calling again. Because of this a small number of interviews were lost this way.

As previously mentioned, it is not always possible to make contact with all members of a household during one call. Since several interviewers may share the interviews in a single household, it was necessary for the calculation of interviewer remuneration to tie information on each interviewer to every single subquestionnaire. It was also important that this item of information could not be changed by an interviewer who was assigned to the remaining interviews.

We solved this problem by defining a question called INTWRID in the subquestionnaire. This question contains an interviewer identification and has the attributes HIDDEN and PROTECT. The value is automatically imputed with the help of a reference to an external file, which in our case is created during the log on procedure and placed in the interviewers private directory.

Since there may be more than one interview in every household the built-in reporting facilities in CATI-Management cannot be used to get an updated review of the number of completed interviews. It was, however, an easy job to produce the relevant tables using Abacus.

4. Concluding remarks

At Danmarks Statistik we have very positive experiences using Blaise to build Data Entry and Validation Systems as well as Interview Systems.

Also the CATI Call Management System has been quite useful when used for person-oriented interview systems like the above mentioned Omnibus Survey.

However, if used for household surveys where the eligible persons of the household are known or selected in advance, we had to make a compromise between our wishes and the restrictions imposed by the system. These restrictions were imposed by the Call Management System (for example the fixed format of the dial screen which considerably limited the information about the household available to the interviewer when dialing), and by the fact that a questionnaire description language (i.e. Blaise) is not intended for building administrative systems.

For the interviewers it is important that a Call Management System appears and reacts in the same way, regardless of whether it is used for person - or household interviews. Therefore, our needs would be better met by a more flexible administrative system, where a dial screen should lead directly to one or more interviews and thus separating the questionnaire and the administrative system.

NEW FEATURES OF BLAISE

Martje Roessingh and Peter Stegehuis
Netherlands Central Bureau of Statistics, Voorburg

1. Introduction

In this paper three new features in Blaise are described: lists, user functions and procedures and coding with trigrams. Lists and user functions are implemented in version 2.4, coding with trigrams will be available in the autumn of 1992. We start in the next section with a simple example and a short introduction to the use of lists. These lists are simple external files that are stored in normal or extended memory. Consulting lists is therefore much faster than consulting external files.

Section three describes a new way of coding. Coding in Blaise can be done step-wise, with an hierarchical classification, or alphabetical by searching a dictionary. In the version of Blaise that is currently under development a more sophisticated algorithm to search the dictionary is implemented in which trigrams (three-letter combinations) are used to build an index for the vocabulary. Coding with trigrams is not included in version 2.41, so you will not find any documentation on this subject in the *Blaise 2.4 Update Manual.* It will be distributed separately.

The subject of the last section is user functions and user procedures. This subject is also treated in the *Update Manual.* Here a small example is presented.

2. Lists

Lists are an alternative to external files: lists are files that are loaded completely into memory, and therefore searching lists will usually be quicker than searching "normal" external files. Lists are stored in conventional and extended memory. They differ in form from external files: lists are sorted and don't have an index file. The use of lists is similar to the use of external files: you can search and read lists.

There are three different forms of lists: binary, ASCII without separators and ASCII with separators. The records in a list all have the same length and follow each other directly. To make a binary list of an ASCII or a Blaise file is very easy with the program Manipula.

The lists are read into the memory of the computer at the beginning of the program. They are stored in conventional memory or in extended memory, but they cannot be stored in expanded memory. If there is insufficient memory for all lists at the same moment, there will be some shifting and reading at run time. The program tries to keep lists that are used most in memory, but this cannot be garanteed. So reading and shifting may cost a lot of time.

Instead of explaining lists in detail, we will show the use of a list with a simple example.

```
QUESTIONNAIRE Prices;

EXTERNAL
  VAR
    ArtPrice: LIST "ARTPR92.LST" (INTEGER [4], REAL, REAL);

ENDEXTERNAL;

VAR
  LowerBnd, UpperBnd: REAL;

QUEST
  Number: 1..99 (KEY);
  ArtCode "What is the article code": 1..9999;
  ArtPrice "What is the price of the article": 0.00..1000.00;

ROUTE
  Number; ArtCode; ArtPrice;

CHECK
  SEARCHLIST (ArtPrice, ArtCode, ?LowerBnd, ?UpperBnd)
    "The article code does not exist";
  ArtPrice > LowerBnd AND ArtPrice < UpperBnd
    "Price must be between $LowerBnd and $UpperBnd";

ENDQUESTIONNAIRE.
```

A list has to be declared in the external paragraph of the questionnaire. In our example:

```
EXTERNAL
  VAR
    ArtPrice: LIST "ARTPR92.LST" (INTEGER [4], REAL, REAL);

ENDEXTERNAL;
```

ArtPrice is the name that is used later in the questionnaire to refer to the list. On disk the list can be found in the file with the name ARTPR92.LST. This name can also begin with a full path to a directory where the list is stored. INTEGER [4], REAL, REAL means that the records in the list consist of an integer (of 4 bytes) and two reals.

In the example the list is used for two things. Firstly the code of the article (answer to the question Article) must be in the list. So there has to be a record starting with this code. Here this code is an integer. It can be checked this way:

```
CHECK
  SEARCHLIST (ArtPrice, ArtCode)
    "The article code does not exist";
```

SEARCHLIST, like searchfile, is used as a *direct check*: if there is no record beginning with the code an error will occur. In our example we use SEARCHLIST also to read two variables: LowerBnd and UpperBnd, both of type real. To indicate that a variable must be read there is a question mark before this variable. So SEARCHLIST reading the two variables can be formulated like this:

```
  SEARCHLIST (ArtPrice, ArtCode, ?LowerBnd, ?UpperBnd)
```

In another direct check we use those variables to check the price of the article. This is nothing new:

```
  ArtPrice > LowerBnd AND ArtPrice < UpperBnd
    "Price must be between $LowerBnd and $UpperBnd";
```

Besides SEARCHLIST there is READLIST which works analogous to READFILE and READBLAISE. In the *Update Manual* you can find the exact details of the syntax and more examples of the use of lists.

At the CBS lists are in use in a special test version of Blaise since spring 1991. The Department for Foreign Trade Statistics is the most important tester of this feature. It uses lists in several applications, for example in a big questionnaire with only 27 questions but with very complex checks. The questionnaire uses 73 lists, together 1.8 megabytes, and is part of a larger batch system. The batch program starts with some archiving and two conversions and ends with an integral check on all forms. Those steps together take about 17 minutes for 1300 forms, which is less then one second per form.

Lists are a fast alternative to external files, but there are limitations to their sizes, so it is impossible to substitute a list for every external file. In the *Update Manual* you can read more about the use of lists.

3. Coding with trigrams

Since last spring the coders of the CBS survey of family expenditures are testing a new method of coding in Blaise, which uses so called trigrams: three-letter combinations. This new method resembles an alphabetical search, but the list that appears on the screen consists of descriptions that partially match the searched word. The spelling does not have to be correct, since all descriptions that look like this word will be displayed. An advantage to a hierarchical search is that the coder for example does not have to know whether tomatoes are vegetables or fruit.

In this section we begin with a small example showing some screens made with a very small codelist. Later we explain some of the basics of the algorithm of this method.

The dictionary of this example contains 100 descriptions of the budget survey and is taken from a publication in English. When we look for "footwear" using the alphabetical search, the screen will look like this:

```
                  alphabetical list
footwear and finery                                       32
footwear unspecified (age,sex unknown from '88)          324
fruit                                                    123
furnishing unspecified                                   225
furniture                                                221
furniture, upholstery and linen                           22
game and poultry                                         153
garden and flowers                                       214
gas and electricity                                      241
general body care                                        421

Description ? footwear
```

Using the trigram method we get this screen:

```
                      searchlist
men's footwear                                           321
women's footwear                                         322
children's footwear                                      323
footwear and finery                                       32
clothing and footwear                                      3
footwear unspecified (age,sex unknown from '88)          324

Description ? footwear
```

The alphabetical list only shows the two descriptions that start with "footwear", while the other list consists of every description containing this word. If we don't know how to spell "footwear", but instead look for the word "footwhere", we will get the following screen:

```
-----------------enhanced searchlist--------------------
children's footwear                                323
footwear and finery                                32
footwear unspecified (age,sex unknown from '88)    324
men's footwear                                     321
women's footwear                                   322
clothing and footwear                              3
other food products                                17


Description ? footwhere
```

On this screen we find the same descriptions containing "footwear" and also one with "food products". The last one is at the end of the list because it resembles "footwhere" less than the others.

For the coding with trigrams a kind of index is created for all the descriptions in the alphabetical list. The index consists of trigrams with references to the words in which they occur. For example the word "book" is split up in the trigrams " bo", "boo", "ook" and "ok ". So " bo" can be found in the index with a reference to the word "book" among all the references to other descriptions containing the trigram " bo". When a trigram occurs in too many descriptions it will not be placed in the index. In our example this is the case with the trigrams " an", "and" and "nd ", for the word "and" appears in more then 50 descriptions. This threshold can be set by the user when he makes a trigram index.

At the moment a data entry program using trigrams is started, the index will be read into memory (normal or extended). It is also possible to use it from disk but that will slow down the program. When the coder searches a word, it is split in trigrams. The program creates a list of all descriptions with matching trigrams. This list is sorted on the number of matching trigrams. First the descriptions containing at least 50% of the trigrams of the word are shown. If this list is empty the descriptions with 25% or more trigrams are shown. Here again these thresholds can be set by the user. The displayed list is sorted so the most resembling descriptions come first.

A great advantage of the described method is that no knowledge of the subject is needed to create the index. It is also independent of the language you use. There can be some problems if words consist only of trigrams that are so common that they are not in the index. In that case the coder will have to look for a longer word or a combination of words. There is one general rule for all methods of coding in Blaise: the quality of the coding never better than the quality of your dictionary!

4. User functions and user procedures

In version 2.4 it is possible to use your own Pascal procedures and functions in Blaise data entry programs. User procedures give you complete control over the way you ask questions and plenty of possibilities to influence the layout of the screen. They are meant for those situations where you want to supply your own way of handling a question or a set of questions. For instance, showing a picture on the screen is not possible in Blaise, so if you want the respondent to look at a picture before answering, you can now use your own Pascal procedure.

You can also define your own functions. Like the known functions in Blaise, they have a name, take parameters and return a result. They can then be used in your Blaise program just like the existing standard functions (e.g. SUM, SQRT, JULIAN).

User procedures work on blocks. You have to give the attribute EXTERN to such a block to indicate that your procedure will take over questioning for this block. Your procedure has complete control over the way in which the questions are asked and has the responsibility to return proper answers to the Blaise program. This shows immediately how powerful this new feature is, and how dangerous. You can do many things that weren't possible in Blaise before, positive and negative.

To show the usage of user functions and procedures, a simple example of a questionnaire with a Pascal unit. The unit contains one function and one procedure:

```
Unit MyOwn;

Interface
Uses DOS, CRT;
Function Age (BirthDat: String): LongInt;
Procedure MyQuest (EntryChar: Char; QuestNr: Word;
                   var Answers: String);
Implementation

Function Age (BirthDat: String): LongInt;
var
 DyBrth, MthBrth, YrBrth, res: Integer;
 DyNow, MthNow, YrNow, DayofWk : Word;
begin
 GetDate (YrNow, MthNow, DyNow, DayofWk);
 VAL (Copy (Birthdat, 1, 2), DyBrth, res);
 VAL (Copy (Birthdat, 4, 2), MthBrth, res);
 VAL (Copy (Birthdat, 7, 4), YrBrth, res);
 if (MthNow < MthBrth) or
    ((MthNow = MthBrth) and (DyNow < DyBrth))
  then Age := YrNow - YrBrth - 1
  else Age := YrNow - YrBrth;
end; {Age}

Procedure MyQuest (EntryChar: Char; QuestNr: Word;
                   Var Answers: String);
var
 Answer1, Answer2: String;
begin
 Answer1 :=Copy (Answers, 1, 20); Answer2 :=Copy (Answers, 21, 40);
 GotoXY (1, 24); ClrEol;
 Write ('What is your name?'); GotoXY (26, 24); Write (Answer1);
 GotoXY (26, 24); Readln (Answer1);
 if Length (Answer1) < 20
 then Answer1 :=Answer1 + '                    ';
 Answer1 := Copy (Answer1, 1, 20);
 GotoXY (26, 24); ClrEol; GotoXY (26, 24); Write (Answer1);
 GotoXY (1, 25); ClrEol;
 Write ('What is your profession? '); GotoXY (26,25); Write (Answer2);
 GotoXY (26, 25); Readln (Answer2);
 Answers := Answer1 + Answer2;
end; {MyQuest}

end.
```

Save this unit as MYOWN.PAS. The questionnaire could be:

```
QUESTIONNAIRE Example;

USES "MyOwn.Pas";
NOTEPAD = 4;

BLOCK MyQuest (EXTERN);
QUEST
  Name   "What is your name?": STRING[20];
  Profes "Please give a short description of
          your profession": STRING[40] (EMPTY);
ROUTE
  Name; Profes
ENDBLOCK; {MyQuest}

QUEST
  SeqNum  "Sequence number of this interview?": 1..10000 (KEY);
  NameProf: MyQuest;
  BrthDat "What is your birth date?": DateType;
  YourAge : 0..120 (PROTECT);
  MarStat "What is your marital status?":
           (Married, Notmar "Not married");
ROUTE
  SeqNum; NameProf; BrthDat; YourAge; MarStat

CHECK
  IF BrthDat = RESPONSE "" THEN
    JULIAN (SysDate) - JULIAN (BrthDat) > 0
    "A birthdate cannot be a future date!";
    YR (BrthDat) >= YR (SYSDATE) - 120
    "A person cannot be that old!";
    COMPUTE BrthDat := STDATE (BrthDat);
    COMPUTE YourAge := AGE (BrthDat);
  ENDIF
ENDQUESTIONNAIRE.
```

To be able to use the unit MYOWN.PAS in the questionnaire, you have to state

```
USES "MyOwn.Pas";
```

in the settings section. Note that the block in the questionnaire has the same name as the procedure in the unit and that it has the attribute EXTERN.

You can now check the syntax of the questionnaire and compile it. During the syntax check the file(s) mentioned after USES must be present (in this example MYOWN.PAS). During compilation either MYOWN.PAS or MYOWN.TPU must be present.

While interviewing, first the key question is asked and then instead of asking the questions in the block the program calls the external procedure to get answers to the questions Name and Profes. The answers to these questions are passed on from the Blaise program to the procedure, concatenated in one string Answers. When first called this string is still empty. In the procedure you can see that the Answers-string is 'unpacked' so you can work with the separate answers.

Then the question "What is your name?" is asked on line 24 of the screen and you can type the answer on this line. The answer is stored in the string Answer1, which is then brought to its proper length (twenty characters). Next the question "What is your profession?" is asked on line 25, and again you are prompted for an answer on the same line. This answer is stored in the string Answer2. At the end of the procedure the strings Answer1 and Answer2 are concatenated to the string Answers, which returns the new answers to the Blaise program.

It is important that the length of Answer1 is exactly twenty characters, because Blaise takes the first twenty characters as the answer to the question Name and the next forty characters of the Answers-string as the answer to the question Profes.

If you want to change the answer to one of the questions in the EXTERN block, you can go back to one of the questions. The program does not automatically call your procedure, since you may just want to page through your questionnaire. As soon as you start editing the answer, by pressing <F2> or a letter-key, your procedure gets called again. On your screen you see the question text as specified in your Blaise questionnaire, and if you press a key (as if you were editing the answer), your procedure will take over again. Again on line 24 and 25 the two questions are asked, the answers you gave the last time are displayed, and you can change them. The new answers are passed to the Blaise questionnaire again.

The key you pressed to start editing is stored in the variable EntryChar - the first parameter in the heading of the procedure - and can be used in the procedure. For instance, in the Blaise question text you could include: "If you want ..., then press <A>; otherwise press <B>". The second variable in the heading, QuestNr, gives you the possibility to see from which question in the block you 'jumped' to the procedure. If it was the first question in the block (in this example the question Name), then QuestNr gets the value 1, if it was from the second question (here Profes), then QuestNr equals 2. If for instance in our example you start editing the question Profes - the second question in the EXTERN block, then the variable QuestNr can be used to make sure the procedure does not start from the beginning (asking "What is your name?"), but asks only "What is your profession?". In this example the variables EntryChar and QuestNr are not used.

After editing the questions in the EXTERN block, Blaise takes over, storing the given answers in the form. Next the question BrthDat asks for a date of birth. In the check paragraph this date is used to calculate the age of the respondent with the user function Age. This function takes as parameter the answer given to the question BrthDat, and returns the age of the respondent in years, using the system date of the computer for the calculation. The system date is obtained with the standard Pascal function GetDate.

This is obviously a very simple example: you can use the whole screen instead of just two lines, show pictures with the questions, etcetera. If you want to use your own functions or procedures in a questionnaire, you should test it extensively before the survey actually starts. Another small example and some more details about the use of user procedures and user functions can be found in the *Update Manual*.

BLAISE TO ITS LIMITS
Applications in a market research environment

Dirk Sikkel and Mylène van Veldhoven
Research International Nederland

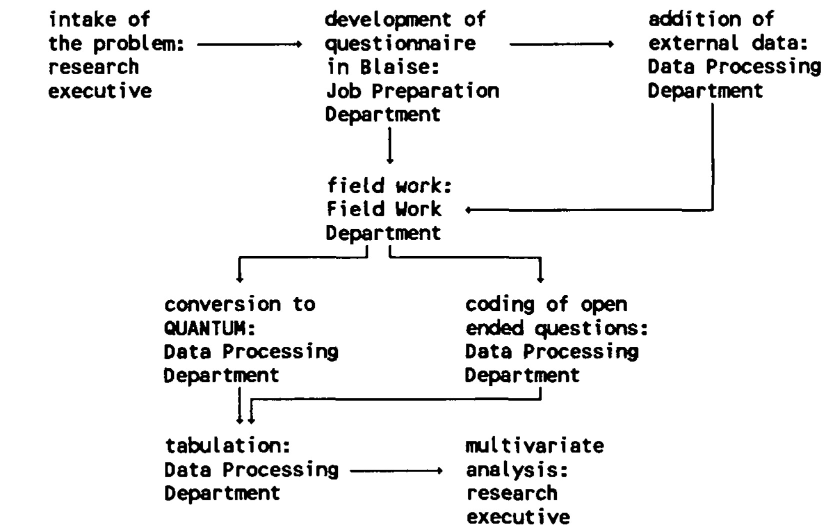
1. Introduction

During the past four years, Blaise has been used by Research International Nederland for a variety of purposes. The reason for buying Blaise was the necessity to carry out CAPI projects that were too complicated for the simple software (Ci2) that was already in use. After the first two projects, which were desperately difficult to manage, Blaise has gradually become a standard tool, embedded in the daily routine of the organization. Although not designed for a market research environment, it enabled us to solve a number of problems which at first sight seemed to be almost insurmountable; as a consequence, the problem solving ability of Blaise gave us considerable edge in some parts of the market. In this paper we will first describe the organization around the use of Blaise. Next some of the problems that have been solved by Blaise will be sketched. Finally we will comment on some of the shortcomings of the package in a market research environment.

2. Organization

The technical units in market research agencies usually are small. Only a few people are involved in the technical development of questionnaires and the subsequent data processing. At Research International, the most common procedure, based on such a small staff, is sketched in figure 1.

Figure 1. The organization around Blaise



Clients who have problems which require complex Blaise questionnaires are dealt with by a research executive with at least some knowledge of Blaise. Next the problem is taken to the Job Preparation Department, where a specialist constructs the actual Blaise questionnaire, usually in close cooperation with the research executive. Sometimes, external data are used in the interview, e.g. in a panel survey. The external data are included in the questionnaire as extra questions. The partly filled records are produced by the data processing department, based on the record descriptions which are automatically generated. After the fieldwork, the data are prepared for tabulation by QUANTUM, a tabulation package which is popular in market research. The interface between Blaise and QUANTUM is rather complex, as QUANTUM is based on 80-column records. The setup generation language of Blaise has, however, enabled us to write a satisfactory conversion program. Open ended questions first are converted to the format of a coding program that is a part of the Ci2 package. After coding these questions, the resulting data are added to the main data set. The SPSS-setup of Blaise is frequently used for preparation for multivariate analysis.

In the whole procedure, usually three people are involved: the research executive, the expert from the job preparation department and the expert from the data processing department. The intake of a job from the client is a difficult process. It is our experience that many clients usually have false expectations of both the possibilities and the impossibilities of computer assisted interviewing. When agreement has been reached with the client, it takes 2 to 4 days to build the (complex) questionnaire in Blaise. After that, the process continues more or less automatically, according to the principles of Integrated Survey Processing.

3. Trade-off processes

An important tool in market research is the trade-off. Respondents have to give preferences for product attributes. After a number of trade-off, utilities of the product attributes are computed which enable the user to construct the 'ideal' or 'optimal' product. This method, for which many specialized CAPI packages are available on the market, is called conjoint analysis. Research International has developed a variant of this technique which is often used for service quality studies. This technique is called SMART (Salient Multi-Attribute Research Technique).

Basically, SMART works as follows. A respondent is confronted with a number of items which are aspects of service quality. A typical example is:

How long do you normally have to wait in a queue?	1. more than 10 minutes 2. 5 to 10 minutes 3. 1 to 5 minutes 4. less than 1 minute

Apart from waiting time, such items deal with friendliness of staff, competence of staff, information, complaints etc. The respondents score one or more companies with respect to these items and indicate the importance of the items. The complicated part (with respect to Blaise) starts where the respondents have selected the eight most importance items. The respondents are presented with an imaginary company where all eight most important items are at their lowest level. So waiting time is more than 10 minutes, the staff unfriendly, incompetent, no information is given etc. The respondent is asked to raise one item one level. This is the most important improvement in this imaginary company. Next the respondent is asked to make a new improvement by raising one item one level. This process continues until all items are raised to their highest levels or no further improvements are desired by the respondents. At every step the respondent is required to make a trade off which improvement has highest priority for him. The resulting data set is analyzed with a tailor made package. Results are mainly in terms of priorities which clients put on service improvement.

Our main interest here is the technical problem of programming this procedure as a CAPI program. A simple solution is to write a dedicated CAPI program. This, however has the drawback that it is hardly possible to make flexible adjustments, according to the project at hand. So it appeared to be necessary to have a basic Blaise version that could be adapted to the needs of each individual client. This was possible, but very complex. The source code of the most complex block (where the improvements are made) is given below.

```
BLOCK RANK_LEVELS;
VAR j, k, y: integer;
    l1: array [1..40] of integer;
QUEST
   rank1 "INTERVIEWER: GIVE THE RESPONDENT THE FOLLOWING CARDS TO ORDER
          THE LEVELS
        ////$t[1]$t[2]$t[3]$t[4]$t[5]$t[6]$t[7]$t[8]$t[9]$t[10]
        $t[11]$t[12]$t[13]$t[14]$t[15]$t[16]$t[17]$t[18]$t[19]$t[20]
        $t[21]$t[22]$t[23]$t[24]$t[25]$t[26]$t[27]$t[28]$t[29]
          //On these cards the most important levels are given. With
          all these aspects, we start at the lowest level. Which aspect
          would you want to raise first. Just name the aspect number.":
   1..40;

   rank "You just raised////$t[y]//to level $l[y].//Of which next aspect
         would you raise the level? Just name the aspect number.
         ////: INTERVIEWER: ENTER ZERO IF THE RESPONDENT DOES NOT WANT
               TO RAISE ANYMORE ASPECTS
         array[2..40] of 0..40;
ROUTE
   for j:=1 to 40 do compute l1[j]:=1 enddo;
   rank1;
   compute y:=rank1; compute l[y]:=l[y]+1;
   rank[2];
   compute y:=rank[2]; compute l[y]:=l[y]+1;
   for j:=3 to 40 do
       if (rank[j-1]<>0) and (j<=n_jumps) then
          rank[j]; compute y:=rank[j]; compute l[y]:=l[y]+1;
       endif;
   enddo;
CHECK
   x[rank1]=2 "card $rank1 is not in the pile of the respondent.";
   compute l1[rank1]:=l1[rank1]+1;
   for j:=2 to 40 do
    if rank[j]<>0 then
      x[rank[j]]=2 "Card $rank[j] is not in the pile of the respondent";
      compute l1[rank[j]]:=l1[rank[j]]+1;
      l1[rank[j]]<=a[rank[j]] "The level of aspect $rank[j] is now
         $l1[rank[j]].//This is impossible as the highest level is
         $a[rank[j]].";
    endif;
   enddo;
```

A program was developed as a part of the integrated SMART-procedure, which generates a standard Blaise source code, given the texts of the items and levels (in the same way Blaise generates SPSS setups). This procedure appeared to be workable. In 50% of the cases the simple dedicated CAPI package is used; in the more complex cases, Blaise performs in a satisfactory way.

4. Public transport

Public transport is an area where many satisfaction studies are carried out. In the Netherlands each larger city has its own public transport company, each of which has unique problems and policies. There are, however, many issues that these companies have in common. The Blaise subfile structure appeared to be very well suited for this type of problem.

The basic questionnaire, common to each of the cities, contained questions like "how often do you travel by bus?", "do you travel at night?", "do you have a driving license?", etc. The answers were stored in the main file. The satisfaction questions were unique to each city; the respondents were asked to score aspects of public transport on a scale between 1 and 10. Such aspects are the shelters, the driving behaviour of the bus driver, the waiting time for the tram, the cleanliness of the subway etc. The evaluations of these aspects were written to a subfile.

A drawback of Blaise is that it does not allow for rotation of questions. In this study, some rotation was required by the client. The problem was solved by rotation based on the respondent number. Four different orders of the satisfaction questions were programmed. They were picked according to the remainder of the respondent number divided by 4 (r mod 4). In the questionnaire this could be calculated as

```
compute order:= r - 4*int(r/4).
```

The ROUTE section then looked as follows

```
if order=0 then ORDER0 endif;
if order=1 then ORDER1 endif;
if order=2 then ORDER2 endif;
if order=3 then ORDER3 endif;
REORDER;
```

The blocks ORDER0, ORDER1, ORDER2, and ORDER3 contain the satisfaction questions in different orders. The block REORDER copies the data into a block of fixed order. By these procedures, using subfiles and reordering the variants to one fixed order, the ease of data processing was optimized within Blaise: each possible variable was written to a fixed field within the record.

5. An incomplete block design

A problem which resembles the order problem in the previous section arose in the context of an advertising monitor. A large bank wanted to evaluate the newspaper ads and TV-commercials of itself and its main competitors. There were six printed advertisements, say a to f and six story boards of TV-commercials, say A to F. From a pilot survey it was clear that it was impossible to have a respondent evaluate all advertisements a to f and A to F. Three from a to f and three from A to F was the maximum. Since there could be an effect of one advertisement on of another, a balanced design was required. This led to 20 combinations (6*5*4/(1*2*3)) like adeBCF or befACD. The 20 combinations were identified by the respondent number or, more in particular, by r mod 20 (the remainder of the respondent number divided by 20).

These respondent numbers were preprogrammed for each interviewer on her field diskette. As a consequence, each interviewer was assigned one combination from a to f and A to F.

6. Sensitive questions

A serious problem in survey research is the inclination of respondents to refuse to answer sensitive questions. In Blaise there is a (hidden) opportunity to enhance the privacy of the respondents when such questions have to be answered. Of course, the interviewer has the possibility to have the respondent enter the answer personally, without the interviewer watching. In principle, by paging backwards, the interviewer can read back the sensitive answer. In the following example this is prevented.

```
TYPE
   IncType = (V0T16 "0 to 1650 guilders",
              V16T22 "1650 to 2200 guilders",
              V22T32 "2200 tot 3200 guilders",
              M32 "more than 3200 guilders",
              NA "no answer");
QUEST
   IncSec "INTERVIEWER: SHOW CARD 20 'TOTAL INCOME OF HOUSEHOLD PER
      MONTH'.////Now I would like you to indicate to which class the
      total monthly income of your household belongs. If you don't want
      to say that to me you can enter it yourself. Afterwards, I cannot
      see what you have entered but it is registered with the data that
      will be processed anonymously.////INTERVIEWER: THE RESPONDENT MAY
      ENTER HIS INCOME IF HE WANTS TO":
   IncType;
   IncReal: IncType (hidden); ROUTE
   page; IncSec; IncReal; page; SIGNAL
   if IncSec<>NA then
       compute IncReal:=IncSec;
       compute IncSec:=NA;
   endif;
```

The procedure works as follows: as soon as an answer is given to the question IncSec (secret income) it is copied to the hidden question IncReal (real income). Then IncSec is reset to the answer NA, which is all the interviewer can see when the computer is returned to her. Our experience is that this procedure reduces item non response. The drawback, however, may be that lack of supervision by the interviewer may lead to non serious answers.

7. A marketing instrument

Sometimes, a Blaise questionnaire can be so convincing, that it can be used for other purposes. The problem under investigation was the question whether a new insurance product should be put on the market or not. To appreciate the benefits of the product, it is necessary to have some insight into the costs of cars. From earlier research it was known that consumers do not like to be confronted with the monthly costs of their car. This happened in paper and pencil interviews where a rough estimate was made, based on the price of the car. In general, respondents are unpleasantly surprised by the height of this estimate and usually deny it. This experience was a good reason to doubt the profitability of the test product.

To investigate the problem, a short but complicated Blaise questionnaire was constructed. The questionnaire started with the price of the car. Next, the costs of the car were broken down to depreciation (yearly), fuel (monthly), tax (yearly), insurance (yearly) and maintenance (monthly). After entering the answers, a summary screen appeared in which the total monthly costs were calculated, visibly broken down to its components. Moreover it was indicated whether the costs were plausible, given the price of the car. When the costs were implausible, the interviewer went over the components for a second time. The results were clear for the respondent and caused no irritation. This clarity of the costs made it easy to introduce the new product and show its benefits. As a result, the product was accepted by the majority of the respondents.

The result of the project was not only a clear positive advice to launch the new product, unexpectedly the questionnaire also showed how to sell it. Presently a dedicated program is being developed which has to be used by the salespeople of the client on laptop computers. It is based on the Blaise questionnaire. From information of the current car of the consumer and usage and insurance data, the benefits of the new product are convincingly calculated and presented.

8. The scenario approach

The most advanced project, or rather method, based on Blaise is the scenario approach. This method makes use of the programming flexibility of Blaise by breaking down a complex consumer decision into a number of detailed separate steps. Here we give an example of mortgages.

The problem starts with a paradox. When respondents are asked in a superficial way why they have taken a particular mortgage they usually answer that the interest rate was the main argument to go to a particular bank or other financial institute (for convenience we will only talk of banks). This, however, cannot be true, for the vast majority of respondents compare only two or three banks with respect to anything. So necessarily, all other banks are excluded for other reasons. In order to investigate this problem, the following sets were distinguished.

A = all banks

B = all banks known by the respondent

C = all banks not refused *a priori* for general reasons

D = all banks not refused *a priori* because of their mortgage products

E = all banks where the respondent has asked for information

F = the bank where the mortgage was effected

G = the banks where the respondent was a client

The sets A through F are ordered, i.e. A⊃B⊃C⊃D⊃E⊃F. Of all banks in the sets B\C, C, C\D, D, D\E, E, E\G and G\E questions were asked why they were in that particular set. With each of these sets, different questions are associated, e.g. for the set B\C:

```
NegBank "INTERVIEWER: SHOW CARD 5.////Can you tell me why you don't
         want to have nothing to do with $MortBank[j] a priori?":
array [1..20] of set[5] of
(Number "they treat you like a number",
 NotKnow "I don't know this bank enough",
 UnFriend "they treat you unfriendly",
 Rich "it is a bank for rich people",
 OldFash "old-fashioned institute",
 NegImpr "I don't know exactly, but I have a negative impression of
         this bank",
 Squeeze "they try to squeeze as much money as possible out of you",
 Business "a bank for business people",
 UnPers "you are treated unpersonally",
 DisTrust "I don't trust this bank enough",
 TooSmall "the bank is too small",
 TooFar "the bank is too far away",
 NotExp "insufficient expertise",
 InConsid "they are too inconsiderate",
 Expens "they are rather expensive",
 IntCli "they are unaware of the interests of their clients",
 Black "black money, ties with South-Africa, etc.",
 Stories "I heard all kinds of negative stories about this bank",
 Oth "other");
```

In order to keep track of the sets, and the associated questions to be asked, indicator arrays were defined, like

I_E {bank indicators in set E}: array[1..20] of integer;

if I_E[k]=1 then bank k is in set E; if I_E[k]=0, then it is not in set E. The final step (from E to F) was analyzed by asking exactly the issues on which the banks in E were compared and the pros and cons of each of the banks. The final outcome of this process was a completely new picture of what is important for marketing mortgages in which interest rate has its own, rather modest, place.

9. Problems and prospects

So far, the experiences with Blaise which are described in this paper are success stories; lack of space prevents us to describe more of them. There are, however, also problems with the package, sometimes due to the basic structure, sometimes due to the neglect of typical market research requirements. The main problems are: lack of speed (especially in questionnaires like in section 8), no easy possibility for rotation of both questions and answers, no possibility of masking answers in set-questions, no easy way of addressing set-response categories by for-loops, no checks for sample quota, no easy interface (including data sharing) with external programs. Although Blaise allows for larger questionnaires than most of its competitors, it is unclear when Blaise reaches its limits.

THE ROLE AND POSITION OF THE BLAISE SYSTEM IN THE SERBIAN STATISTICAL OFFICE

*Milorad Stojanović *)*
Serbian Statistical Office, Beograd, Yugoslavia

The Serbian Statistical Office (SSO) is the official statistical office in the Republic of Serbia in Yugoslavia. It is a governmental organization. It is organized in a decentralized way on territory of Serbia with headquarters in Belgrade and regional offices throughout Serbia, including Vojvodina and Kosovo, and a city statistical office in Belgrade (for the territory of Belgrade).

We have quite different equipment for data processing in various parts of our office. In our offices in Vojvodina, we have been using since 1989 Unisys microcomputers with the Application Development System (ADS) and a very old IBM S/1. Our city statistical office in Belgrade has a huge IBM mainframe and some Nixdorf microcomputers for data collecting. At the SSO headquarters in Belgrade we are using an IBM 3090 model 17T with MVS/ESA, TSO and our proprietary software package GODAR for data collecting, correcting and managing of data processing by statisticians, and an IBM 3742 Data Entry Machine for huge data input. Outside Beldgrade we have used IBM 5280 Distributed Data Systems since 1983 with tailor made software (written in the DE/RPG language). In Kosovo we use a cluster of terminals connected to the mainframe in Belgrade and an IBM S/1. The mainframes in Belgrade (two) and in Vojvodina (one) are connected.

This year we started replacing our technologically obsolete equipment (IBM 5280, IBM 3742 and IBM S/1) by PC's with the Blaise System. In our headquarters in Belgrade we shall replace all our IBM 3742 machines by PC-LAN's with Novell Netware and the Blaise System. In the headquarters, professional key punching clercks will key in a massive amount

*) The views expressed in this paper are those of the author and do not necessarily reflect the policies of the Serbian Statistical Office.

of data; after that statisticians will clean and verify those data. In our regional offices we shall put PC's with the Blaise System as stand alone systems and in some near future we plan small LAN's. All our regional offices will be connected via gateways and leased lines to the mainframe in Belgrade.

During the first phase in our regional offices we shall produce only clean data and preliminary results.

The role of Blaise, in the first phase, will be to support data collecting, controlling and correcting on the level of the questionnaire, tabulation of the 'first' results and transferring data to the headquarters.

The position of the Blaise System in SSO as a whole will be as a secondary system, but in our regional offices and in Kosovo it will be the only system.

Until now, we have developed a few applications. The first one is from the statistics of agriculture about early cereals and industrial crops. The others are from the vital statistics, namely from the statistics of natality and statistics of mortality.

To automate the whole process our colleagues have developed a skeleton program or shell, written in Pascal, which enables end users with a minimum of knowledge of PC's and MS-DOS to:

1. communicate with the PC
2. choose the appropriate area of statistics
3. choose the statistical survey
4. type in data and to control them
5. correct data
6. tabulate - produce 'first' results
7. import/export programs and/or data files from/to SSO in Belgrade
8. produce backup files

This system of menus enables end users to use in a very efficient way the statistical applications they need, under the Blaise System.

In the meantime, we have used Blaise in two different areas.

During the elections in '92, we have developed CAPI programs for data collecting, controlling and correcting and a few Pascal programs for tabulating the results of the voting. We were very satisfied with Blaise.

The second application, but as a pilot version only, was the use of Blaise for the application of refugees. In that questionnaire were several (eight) coding questions. The whole process of coding was very quick, easy and accurate.

In usage of Blaise we found some 'problems'. In the first place, in our proprietary system GODAR criterions for logical and numerical control must be done using 'negative' logic. Blaise needs 'positive' logic.

Secondly, when codes exceed 10 digits, in coding files, the system automatically truncates the eleventh and other digits to the right.

Thirdly, in our country we use a Modulo-11 function, but with a quite different algorithm. We like to know if it is possible to incorporate our Modulo-11, through some Pascal function, in Blaise.

We like to say only one thing on behalf of the Blaise System - IT IS OUR DECISION FOR THE NEXT SEVERAL YEARS.

As conclusion the main change in our daily work shall be greater involvement of statisticians in regional offices in the automation process of all phases of their work, from the collecting of questionnaires until the end - producing statistical tables ('first' results), under their own supervision.

SURVEYS TO EVALUATE BRITISH RAIL'S ROLLING STOCK

George Terzis and Geoff Copley
The MVA Consultancy, Woking, England

1. Introduction

The aim of this paper is to describe the context in which The MVA Consultancy applies the Blaise software to quantitative surveys. In particular, we discuss in some detail a recent application carried out for InterCity to assess demand for rail travel in relation to different types of rolling stock.

InterCity (IC) is one of British Rail's (BR) four passenger businesses mainly catering for medium and long distance travel in England and Wales. It operates to a commercial, profit making remit, with a turnover of around £830 million and in 1990/91 a trading profit of £50 million.

The IC Mark IV stock has been introduced on the East Coast Main Line since 1989. In 1991 the service was established on this route and most passengers had experienced both Mark IV and Mark III (which is also in service on this route) stock. IC required a comprehensive study to understand the perceived differences between the two types of carriages.

The MVA Consultancy was commissioned to carry out research with the following objectives:

- to obtain quantitative evidence of the differences between the two types of carriages as perceived by passengers;

- to isolate the stock effects from journey time improvements and to give overall and detailed valuations of the carriages aspects (appearance, seating, ride quality, etc).

The tight timescale of the study (three months to report) and the decision to customise the interview to suit each respondent's circumstances and perceptions required the use of Computer Assisted Personal Interviews (CAPI). In general in the field of transport research, the advantages of CAPI relate to speed of data collection but also most importantly to increased accuracy of the results and enhanced ability to elicit complex information.

2. Methodology

Usually such research is preceded by a qualitative phase with in-depth interviews, either in group discussions or personal semi-structured interviews on trains, in order to understand the areas of interest and define the relevant attributes. In this case this was not necessary because of extensive past experience and a number of qualitative and quantitative surveys about BR's rolling stock. In particular, The MVA Consultancy had undertaken a similar study in 1985, with a substantial qualitative stage, to assess the differences between the Mark III carriage and its predecessor Mark II (at that time pen and paper interview (PAPI) techniques were used for the main fieldwork). In addition, a number of studies in the UK and in the Netherlands provide enough empirical evidence about the most important attributes of carriage quality.

It was decided to include the following seven aspects of carriage quality in the assessment:

- luggage;
- appearance and cleanliness;
- ride quality;
- seating;
- access/egress;
- toilet facilities;
- environment control.

A questionnaire was designed to collect information in the following areas in broad terms:

- passenger and journey details to customise interviews;
- the ranked order of importance of the seven carriage quality aspects;
- the perceived quality levels for each aspect of the carriage where the interview took place;
- the satisfaction level for each aspect;
- the existence of noticeable differences between Mark III and Mark IV stock;
- the value of these noticeable differences using stated preference techniques.

The fieldwork was conducted on BR trains with lap top computers and more than 600 passengers were interviewed. They were recruited on either Mark III or Mark IV carriages on the East Coast Main Line. The computerised interview was facilitated by other visual aids including artist's impressions of the different levels of aspects of carriages. These artist's impressions were instrumental in introducing the subject of the research. They also helped the interviewee's concentration in carrying out ranking exercises, about attitudes and satisfactions of stock, customised to her/his perceptions. The respondents were asked to compare the carriage they were travelling in with the other type of stock. In addition to the artist's impressions of the stock details, photographs of each carriage were also shown.

A series of questions was incorporated concerning the quality of the carriage. The artist's impressions were used to help the respondent indicate the level which best described the quality of each aspect. The degree of satisfaction each respondent felt for the perceived level of quality was recorded on a five point scale, defined as: very good, good, adequate, poor and very poor.

Perceived differences between the two types of stock were noted on each quality aspect. Given that a difference on one of the aspects had been acknowledged, the questionnaire explored further differences at a more detailed level. For example, if a respondent indicated that there was a seating difference between Mark III and Mark IV, then the respondent was asked if the difference was in the following:

- comfort of the seat;
- adjustment of the seat;
- legroom;
- the seating layout.

In some cases more than one difference was perceived. All seven main aspects of carriage quality were broken down into their important component parts. Each difference was recorded using the five point scale below to show which type of carriage was preferred:

- definitely prefer $current carriage;
- slightly prefer $current carriage;
- no difference;
- slightly prefer $other carriage;
- definitely prefer $other carriage.

where $current and $other were defined as MarkIII and MarkIV as appropriate.

With the respondent thinking in terms of the differences in rolling stock quality, the two Stated Preference (SP) experiments were introduced. SP techniques are used in transport modelling as an extension of 'revealed preference' (RP) methods. Traditionally, demand analysis is based on RP data; that is choices and decisions that have actually been made in the marketplace. With theoretical advances in econometrics, there has been great expansion in this field. But there are practical limitations to the RP approach, largely connected with survey costs and the difficulty of distinguishing the effects of attributes such as quality or convenience. In addition, for obvious reasons, models based on RP cannot handle new alternatives that might be introduced into the market.

As a consequence, a practice has developed of basing demand estimates on an analysis of responses to hypothetical choices. The alternatives are presented to respondents in terms of their component attributes described at a number of levels. The design of these SP techniques is based on the principles of experimental statistics. Traditionally the presentation of the alternative choices in SP surveys has been carried out using cards and paper. The recent dramatic changes in the micro-computer hardware and software market have facilitated new developments in SP techniques. By using computers, choices can be customised to suit the respondent's circumstances.

In this study we presented two SP experiments to the respondents to assess their valuations of rolling stock, overall, and then discriminate between the various components of the quality of rolling stock. In the first experiment, the respondents were asked to rank a number of different BR services which included trade-offs between return fare, one-way journey time and type of rolling stock. The latter was defined at two levels, Mark III and Mark IV. Both return fare and journey time had three levels, the middle level representing the characteristics of the current journey and the other two a reduction and an increase in fare and journey time.

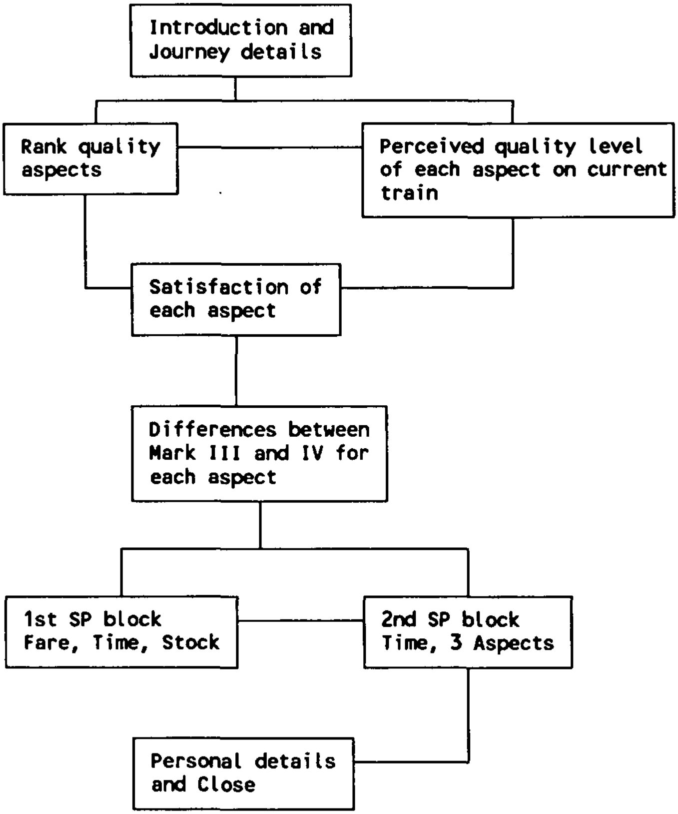
The full factorial design (all attributes presented at all levels in all combinations) for the above variables and levels includes (3x3x2=) 18 options. It was decided to present a total of 16 options to all respondents, but in blocks. Since four options can be clearly presented on each screen, it was decided to divide the design into four sets of four options to trade-off.

The second experiment presented trade-offs between services described in terms of journey time and three specific attributes of carriages. These qualitative attributes were selected for each interviewee according to his importance ratings and perceptions of change. In total the full factorial of the second SP includes (3x2x2x2=) 24 options, of which 12 were presented to the respondents in three sets of four options. The respondents were asked to rank the four options within each set in order of importance. Both the SP experiments were customised to suit current individual circumstances so that realistic questions could be asked.

3. Application

We usually start questionnaire design by drawing a block diagram of questions and groups of questions to be included in the research (see figure 1). After the decision is taken about the basic contents of the research and its feasibility is assessed, the questionnaire is directly designed in the Blaise language and the specification is stored in the computer. In practice, substantial parts of a questionnaire are readily drawn from a library of questions and blocks which we have built up in three years' experience with Blaise.

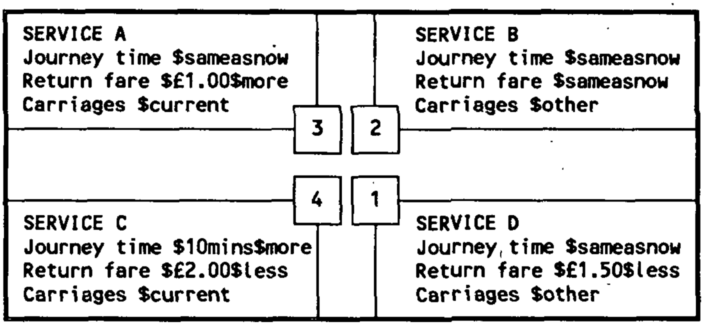
Figure 1. Structure of the questionnaire



Blaise 2.22 was used in this application. The questionnaire included a main section with questions about journey and personal details. Rankings and satisfactions of the quality of carriages and the SP experiments were included in separate blocks. In the main section individual details were established which were used to customise later questions. It was important to establish the journey time and the purpose of travel because the SP designs were customised on the basis of this information.

Extensive use of array questions and array variables was made in the rankings and satisfactions block. An important consideration was to show the respondents their own rankings and allow them to reconsider. In the SP exercises extensive use of substitution enabled the design to be customised for the different types of travellers. As an example we show in figure 2 one set of four options that respondents were asked to choose from in the first SP. The Blaise application is shown at the end of the paper.

Figure 2. Presentation of the first Stated Preference design



In the application of the second SP experiment the following criteria were used to decide which of the seven attributes should be included in the trade-offs:

- if the respondent recognised three or more differences between the two types of stock, then the three ranked most important were incorporated into the SP;

- if the respondent recognised only two different aspects, then these two variables were joined in the SP by the highest ranked of the other five quality aspects;

- if the respondent recognised only one different aspect of carriage quality, then the two highest ranked of the other six were also incorporated into the SP;

- if no aspect was acknowledged to be different between the two types of stock, then the respondent was asked to trade-off the three highest ranked carriage quality aspects.

These three variables were presented as in Mark III and Mark IV carriages and the fourth variable to trade-off was always one-way journey time, at three levels. It was customised to suit the circumstances of each passenger's journey (e.g. see figure 3). The above criteria were implemented into Blaise by means of a series of loops and conditional statements in the *Route* paragraph. Finally, the three variables of quality of carriages shown to each respondent were stored as hidden questions.

Figure 3. Presentation of the second Stated Preference design

```
SERVICE A                              SERVICE B
Journey time $15min$less               Journey time $sameasnow
$Attribute A: Mark IV                  $Attribute A: Mark III
$Attribute B: Mark III                 $Attribute B: Mark IV
$Attribute C: Mark III                 $Attribute C: Mark IV
                         4      1

                         3      2
SERVICE C                              SERVICE D
Journey time $15min$less               Journey time $10min$more
$Attribute A: Mark III                 $Attribute A: Mark III
$Attribute B: Mark III                 $Attribute B: Mark IV
$Attribute C: Mark IV                  $Attribute C: Mark IV
```

An important aspect of our Blaise applications is the ability to check each interview for consistency interactively. For example routing is different depending on the type of passenger being interviewed (eg business and leisure). Most of the common type of interviewers' errors can be avoided (eg answer out of range, origin of journey same as destination, etc). In addition, calculations are carried out by the computer to reduce the interviewers' (and interviewees') burden. In the particular application described in this paper, the computer guided the interviewer to ask and enter the correct departure and arrival time of the journey. Then journey time was calculated and shown to the interviewee together with all the other particular details of his journey.

In the application of the SP experiments the checking facilities were particularly useful. It is a common error in PAPI approaches to rank the same option twice. In this case, this was prevented with a few lines of code (conditional and loop check) in the *Check* paragraph. Another common cause of attrition in the final sample for SP experiments is irrational ranking of options. In order to avoid correlation in the design some cases of dominance are left in the SP design. Thus, some options are always worse, or better than others. If the respondent fails to understand this, the recorded information is useless at the analysis stage. Our approach here is to use either a soft check (*Signal* paragraph), or warnings in the *Route* paragraph to notify both interviewer and interviewee about the error. They are encouraged to go back and examine the rationality of their answers and correct their rankings. It is generally acknowledged that these types of ranking errors can lead to an attrition of 10% to 15% at the analysis stage. Our experience shows that by using the above CAPI techniques attrition can be eliminated or reduced to less than 5%.

4. Discussion

Towards the end of the 1980s The MVA Consultancy evaluated a number of CAPI systems. The decision was taken at the end of 1988 to use Blaise which is at the forefront of CAPI software.

Since then a number of surveys have been written and administered in Blaise by the MVA Consultancy in the fields of market research and passenger and freight transport demand assessment, with considerable success. Beyond doubt we are, in general, satisfied by the software in its present form. However, we are also eagerly looking to further developments to accommodate the evolving nature of our research. We welcome the opportunity to discuss different aspects of Blaise with other practitioners and academics and present our views on current limitations and areas for further development, examples of which are shown below.

- ability to call from Blaise program modules written by users in PASCAL;
- format and screen layout facilities;
- compile big CAPI questionnaires;
- facility to change order of questions and enumerated answers at random;
- more algorithmic and mathematical facilities (nested do loops in route paragraph, additional functions etc).

12
APPENDIX. Example of a Stated Preference design in Blaise

```
Block Spblock "Sp Cards" (Subfile);

Var
   Niter, Oflag1, Oflag2, Oflag3,Iter, Flag: Integer;
   Catch1, Catch2: Integer;
   CardOrder: Array[1..4] of Integer;
   Ch, Last1Rank, Last2Rank, Last3Rank, Last4Rank: Integer;
   Lasx1Rank, Lasx2Rank, Lasx3Rank: Integer;
   RanCh: Array[1..4] of String[6];
   RanFil: String[6];
   jti1, jti3, jtx1, jtx3: Integer;
   jf1, jf2, jf3: Real;
   far, frc, jtime: Real;
   rou: Integer;
   fact1, fact3: Real;

Quest
      SpIntro
"////In this part of this questionnaire we want you, as a rail
traveller, to consider a series of imaginary choices.
//// In each case, please imagine that you were given a choice
//// between FOUR different types of services, for the part of
//// your journey that you are actually making on THIS train
//// today//": Pseudo (Empty, Screen);

      BRack
"////                              PLEASE NOTE
//////The services are entirely imaginary and only set up for research
purposes to assess preferences for the Mark III and Mark IV
carriages.//":
      Pseudo (Empty, Screen);

      DesShow
"//The different services which we will ask you to choose between are
each described in terms of

////RETURN FARE which is either   ONE WAY JOURNEY TIME which is
   /// * 'same as now'                  * 'same as now'
  // OR * a reduction as shown       OR * a reduction as shown
  // OR * an increase as shown       OR * an increase as shown

      ////And finally CARRIAGES which are either
      ////     * Mark III OR
      //       * Mark IV
      //": Pseudo (Empty, Screen);
```

```
     SameOthers
"//////Apart from return fare, journey time and carriages, you can
assume that the journey is exactly the same as the one you are making
today//": Pseudo (Empty, Screen);

     ChoosePlease
  "////  Suppose you could choose between different services
  //// for the part of your journey that you are making on THIS
  //// train today. Please indicate which of the four services
  //// (A, B, C and D) you would most prefer, which would be
  //// your second and third choices, etc//":
       Pseudo (Empty, Screen);

     Des1
     "////Which of the following is your $RanFil choice?//": Array[1..4]
     of (A "     Service A
//return fare:         £ $jf1 less
//journey time:        $jti1 mins less
//rolling stock:       Mark III
//            * $CardOrder[1] *
//-------------------------",
B "     Service B
//return fare:         £ $jf1 less
//journey time:        same as now
//rolling stock:       Mark IV
//            * $CardOrder[2] *
//-------------------------",
C "     Service C
//return fare:         same as now
//journey time:        $jti1 mins less
//rolling stock:       Mark IV
//            * $CardOrder[3] *
//-------------------------",
D "     Service D
//return fare:         £ $jf3 more
//journey time:        $jti1 mins less
//rolling stock:       Mark IV
//            * $CardOrder[4] *
//-------------------------");

     Alarm
 "////          A R E   Y O U   S U R E ?
 //////P L E A S E  G O  B A C K  A N D  C H A N G E  R A N K S
 //// Service  D  SHOULD NOT  be better than service  C
 ////They are similar in journey time and rolling stock but Service D is
more expensive//":
     (Warn "Please use <Up Arrow> to go back and change") (Empty,Screen);
```

```
     Des1Sure
     "//These are your rankings for the first set of services
     //                                  ARE YOU SURE ?//":
     (A "     Service A
//return fare:        £ $jf1 less
//journey time:       $jti1 mins less
//rolling stock:      Mark III
//     RANK  * $CardOrder[1] *
//-------------------------",
B "     Service B
//return fare:        £ $jf1 less
//journey time:       same as now
//rolling stock:      Mark IV
//     RANK  * $CardOrder[2] *
//-------------------------",
C "     Service C
//return fare:        same as now
//journey time:       $jti1 mins less
//rolling stock:      Mark IV
//     RANK  * $CardOrder[3] *
//-------------------------",
D "     Service D
//return fare:        £ $jf3 more
//journey time:       $jti1 mins less
//rolling stock:      Mark IV
//     RANK  * $CardOrder[4] *
//-------------------------") (Empty, Screen);

.................................................................

 define imputed questions for store }
llr1: 1..4 (Hidden);
llr2: 1..4 (Hidden);
llr3: 1..4 (Hidden);
llr4: 1..4 (Hidden);

jjff1: 0.00..50.00 (Hidden);
jjff3: 0.00..50.00 (Hidden);

jjtt1: 0..90 (Hidden);
jjtt3: 0..90 (Hidden);

QChoice: Array[1..3] of String[22] (Hidden);

jjxt1: 0..90 (Hidden);
jjxt3: 0..90 (Hidden);
..........................................................
```

```
Route

{ data statement for choice fill }
     Compute RanCh[1]:= 'FIRST';
     Compute RanCh[2]:= 'SECOND';
     Compute RanCh[3]:= 'THIRD';

     SpIntro; BRack; DesShow; SameOthers; ChoosePlease;

     Compute jf2:= Fare;
     If ((Typeticket = stas) or (Typeticket = frss)) then
       Compute jf2:= 2* Fare;
     endif;

     If (Class = fi) then
       Compute fact1:= 0.1276;
       Compute fact3:= 0.064;
     else
       Compute fact1:= 0.111;
       Compute fact3:= 0.074;
       If (Purpose = emp) then
         Compute fact1:= 0.125;
         Compute fact3:= 0.09375;
       endif;
     endif;

     Compute Far:= fact1 * jf2;
     Compute frc:= Far * 2;
     Compute rou:= Round(frc);
     Compute jf1:= rou / 2;
     If (jf1 < 0.50) then Compute jf1:= 0.50; endif;

     Compute Far:= fact3 * jf2;
     Compute frc:= Far * 2;
     Compute rou:= Round(frc);
     Compute jf3:= rou / 2;
     If (jf3 = 0) then Compute jf3:= 0.25; endif;

     Compute jtime:= 0.136 * Diff;
     Compute frc:= jtime/5;
     Compute rou:= Round(frc);
     Compute jti1:= rou * 5;
     If (jti1 > 50) then Compute jti1:= 50; endif;

     Compute jtime:= 0.227 * Diff;
     Compute frc:= jtime/5;
     Compute rou:= Round(frc);
     Compute jti3:= rou * 5;
     If (jti3 > 60) then Compute jti3:= 60; endif;
```

```
{ ask hidden questions for level values of design }
     jjtt1; jjtt3;
     jjff1; jjff3;

{ start rankings }
     Compute Flag:= 0;
     For Iter:= 1 to 4 do
       If (Iter = 4) then
         Compute Flag:= 1;
         Compute Last1Rank:=10-(ord(Des1[1])+ord(Des1[2])+ord(Des1[3]));
         Compute CardOrder[Last1Rank]:= 4;
       endif;
       If (Flag = 0) then
         Compute RanFil:= RanCh[Iter];
         Des1[Iter];
         Compute Ch:= ord(Des1[Iter]);
         Compute CardOrder[Ch]:= Iter;
       endif;
     enddo;

     If (CardOrder[4] < CardOrder[3]) then
       Alarm;
     endif;

     Des1Sure;
{ ask hidden for the one ranked last }
     llr1;

{ ready for second pack of cards }
     For Iter:= 1 to 4 do
       Compute CardOrder[Iter]:= 0;
     enddo;

     Compute Flag:= 0;
     For Iter:= 1 to 4 do
       If (Iter = 4) then
         Compute Flag:= 1;
         Compute Last2Rank:=10-(ord(Des2[1])+ord(Des2[2])+ord(Des2[3]));
         Compute CardOrder[Last2Rank]:= 4;
       endif;
       If (Flag = 0) then
         Compute RanFil:= RanCh[Iter];
         Des2[Iter];
         Compute Ch:= ord(Des2[Iter]);
         Compute CardOrder[Ch]:= Iter;
       endif;
     enddo;
.........................................................

 EndBlock;
```

THE HOUSING CENSUS: A BLAISE APPLICATION IN AN EXTERNAL CONTEXT

Jan Wieringa and Glenn Fokké
Netherlands Central Bureau of Statistics, Voorburg

Abstract

One has come to consider electronic support for filling in questionnaires as a necessary form of help to informants. They stimulate the informant's willingness to collaborate and they increase the reliability of their responses. Nowadays more and more informants come to use computers; in such a context it is an interesting option for the CBS to provide them with data entry software with which to enter the required data. The Blaise system has proved to be a valuable tool for easy and accurate development of such programs.

1. The census

The Housing Census of the CBS has been held since the beginning of the century and its purpose is to evaluate the stock of housing buildings in the Netherlands. In combination with population statistics, these statistics have always played an important part in the housing policy of the Government. The housing stock is also the most important criterion for the allocation of funds to the local authorities. The latest reassessment of the housing stock took place in 1971, in combination with the population census. Today's housing stock figures are based on the 1971 stand, corrected by the mutations that have taken place in the last years. A new reassessment of the housing stock has been started this year and will be completed in 1995. It will take into account houses and housing units, but also recreation dwellings and specialized residences such as monasteries and homes for elderly people. The information will no longer be confined to housing stock sizes: the aim is now to build an address register, which will be a reliable starting-point to maintain housing stock information of a high quality level (Amse, 1992).

2. The processing

To simplify the burden of filling in the questionnaires for the Housing Survey (about 6 million addresses in all), the local authorities were sent a Blaise questionnaire containing the addresses from the Geographical Base Register. This is a file based on PTT information, containing all existing addresses in the Netherlands. The local authorities are requested to fill in, for each address, a code that identifies it as a house, a housing unit, a recreation dwelling or a specialized residence. In the case of specialized residences the capacity also has to be filled in. It is also possible to add missing addresses. On receiving the filled in forms the CBS performs a check of the information provided by the local authorities, executes the necessary corrections and stores the addresses in an ORACLE database.

3. The application

As the program has to run in an external context it was not compiled as CADI (Computer Assisted Data Input), but as CAPI (Computer Assisted Personal Interviewing). CAPI gives more information on screen for 'strangers', and the CBS is certain to receive 'clean' Blaise data files. Fields can be protected against erroneous values. Another advantage is that the Blaise menu is smaller than in a CADI application: it offers only the options to add, to do a key search and to select. A number of texts have been redefined in the Blaise file CAPITEXT.xxx to make the program more housing-survey specific. For instance the option texts of the Blaise menu have been adapted:

Standard text	Adapted text
Interview Examine form Select forms	New Street Search on Postcode Select Streets

The program was complemented with a few Pascal programs: one of them is the main menu, a shell around all the others, see figure 1. The other programs take care of diverse output functions, such as making summaries and converting to ASCII. These programs read directly from the Blaise database, which was made possible by a Pascal read routine designed by the Blaise team specifically for this project.

Figure 1. The main menu

```
C.B.S.            Administrative Housing Census            C.B.S.
                           MAIN MENU

               1. Revise Address/Dwelling Code
               2. Revise Place-name
               3. Send Data
               4. Make Summary
               5. Set Colours
               6. Make ASCII file

Option: 1/2/3/4/5/6 or v/^.   Activate Option: Enter.   Stop: Esc.
```

The alternative approach, sending the Blaise Conversion Module as part of the package and make tailored programs read their data from the ASCII file, would have met the following objections:

- Disk space wastage on the informant's computer;
- Specifically for large municipalities: long processing times due to the extra conversion.

The CBS department responsible for the Housing Census organized a help desk for help and support of the project. The user manual was written with particular care. As the CBS has little experience with designing and implementing software for external users, the system was carefully tested on a number of test sites.

For check purposes the CBS uses a CADI application developed from the same Blaise definition. This is used to submit the incoming data files to an integral check. This program is also used to enter the data filled in on a paper form or on tape.

3.1. The Blaise program

The key of every address is the postcode and the house-number. In the Dutch postcode system the postcode nearly always identifies a street or part of a street. There are only a few cases where one postcode corresponds to two different streets. These exceptions are taken care of with a sequence number added to the key. The screen layout is primarily determined by the file structure. The main screen (figure 2) presents information concerning the street or part of street (pertaining to one postcode).

Figure 2. The Blaise main screen, street (postcode) information

```
BLAISE 2.32  CAPI   DWELLING    Main          street coding
Street name according to PTT conventions

 Use CAPITALS.


(enter text of no more than 17 characters)
_____________________________________________________________
     PlName       APPINGEDAM
     Street       EPPENSSTR
     From            1
     To             29
     PcNum        9902
     PcLet        HA
     Eod          0    Odd
     SeqNum        1
     WbCode       0002

     WRCode       ░
     AdrCount     7

 BROWSE   F1:Help  F2:Change  Escape:Stop
```

The second screen (figure 3) is a list of the different addresses (house-numbers). For the addresses supplied by the CBS the uses can only fill in the fields Code and Number. CBS addresses cannot be deleted by the local authorities. They can, however, use code 9 to specify that a given address is not a habitation. There are empty lines at the end of the form, allowing new addresses to be added. The house-numbers have to fall within the range defined for the postcode - otherwise they are refused.

There is a code field in the main screen, which can be used to simplify the process of entering codes. If a value is entered in this field, it will be supplied as a default in every code field left blank in the second screen. Another field in the main screen displays a count of the addresses in the second screen.

Figure 3. The second Blaise screen: the specific addresses

```
BLAISE 2.32  CAPI   DWELLING    House          street coding
Habitation code

 1. House
 4. Recreation dwelling
 5. Housing unit, private households
 6. Housing unit, public households
 9. Not a habitation

(fill in a number between 1 and 9)
----------------------------------------------------------------
         Strname            CBSnr CBSxtra Housenr Xtra  Code Count

Regel1   EPPENSSTR            1
Regel2   EPPENSSTR            3
Regel3   EPPENSSTR            5
Regel4   EPPENSSTR            7
Regel5   EPPENSSTR            9
Regel6   EPPENSSTR           11
Regel7   EPPENSSTR           29
Regel8   EPPENSSTR
Regel9   EPPENSSTR
Regel10  EPPENSSTR

 BROWSE   F1:Help  F2:Change  Escape:Stop
```

4. Practical aspects

At the end of 1991 the Dutch local authorities were asked whether they would prefer to receive the address lists for the housing survey on paper forms or on a diskette. In many cases we explained what advantages an electronic file has - also for the authorities themselves - on a paper file. We particularly draw their attention to the possibility of reusing the file for local purposes.

The result was quite encouraging: out of the 647 Dutch municipalities 583, about 90% chose for electronic information interchange. Preference for paper forms was in general only expressed in very small places.

In the spring of 1992 the municipalities that had chosen for a diskette received a program disk complete with installation procedure and one or more data disks with a compressed file containing addresses of buildings susceptible of being used for habitation.

The program is accompanied by a manual with a complete description of all the functions of the program and of the installation procedure.

4.1. Users' findings

The program has probably not yet been installed everywhere, so it is not yet possible to give a complete picture of the problems at hand. In fact few problems have been signaled yet. Some problems have occurred during program installation. This sometimes had to do with a shell program running on the computer. The program turned out to hang when running under some shell programs. This problem could be solved by temporarily deactivating the shell program. Once the program did not work after being copied to another PC. This was caused by the fact that some program files are made hidden and thus cannot be copied in an easy way with DOS commands. Sometimes it was the user who did not precisely understand what was supposed to happen. In all cases the problems could be solved with phone support.

The program meets general enthusiasm of the users. It is user-friendly en does not give rise to substantial problems. Questions we are asked by the users are usually of the conceptual kind. Questions about the program itself are generally simple to answer and typically have to do with the choice of a key combination or with the structure of the menu.

4.2. Incoming data files

As for now (August 1992) we have received files from 35 municipalities, and they have been integrally checked with the CADI program. We are pleased to find that, as of course we had expected, that there are very few errors to be found. Time saving is of course one of the great benefits of this way of collecting information: the data are input by the informants themselves, the technical checks are simple and errors are hardly ever there to be detected.

5. Conclusions and recommendations

We may conclude that this technique of information collection functions properly and that Blaise is an excellent tool for developing data entry programs. The reliability of the software is a great advantage. The fact that the software has to run in an unknown context is decisive in the choice between standard software and tailored programs. Development time of tailored programs can grow significantly if one has to take external factors into account.

The combination of Blaise and some additional Pascal tailored programming came out to work very well. The condition, of course, is the availability of a Pascal read routine to access the Blaise database directly. Maybe the option to generate it could be added to the Setup Generator.

For this census we chose for the CAPI option, although it is meant for interviewing, because this type of application makes it easier to control data entry and to give more information on screen. CAPI, however, is not free of drawbacks. One of these drawbacks is that it is not possible to browse the file backwards. Another disadvantage is that one cannot close a Street (=Interview) and go over to the next street without performing 'many' operations. The user sees this as useless delay. Supposing more such applications will be designed in the future, it may be desirable to offer a fourth Blaise application type besides CADI, CAPI and CATI (T=Telephone).

The user manual must, of course, be designed with particular care. This can avoid quite a few telephone calls! Our strategy was to supply the test sites with a summary manual, giving only the most important information. This enabled us to get quite a picture of the information an external user wants to have at his disposal.

Literature

Amse, A.K., 1992, 'Woningtelling', CBS Signaal, June 1992, pp. 5.