Challenges in developing a Longitudinal Survey on Income Dynamics

Brett Martin, Statistics New Zealand

1. Introduction

The development of a longitudinal panel survey is a very complex and challenging process. Statistics New Zealand is currently running a project to evaluate the feasibility of a Longitudinal Survey on Income Dynamics. This paper discusses some of the challenges involved in building the Blaise instruments for the two field tests involved.

For the first time in Statistics New Zealand a large Computer Assisted Interviewing (CAI) instrument has been constructed without creating an associated paper form. New ways of working together to specify and develop the Blaise programs have evolved as the project progressed.

Checking large instruments takes a considerable amount of time and effort. In a longitudinal survey the need to refer to data collected in earlier waves increases the complexity of routing and editing and the amount of validation required. Ongoing changes to questions and edits further increase the workload.

Data obtained from the Longitudinal Survey on Income Dynamics (LSID) field testing programme is being investigated by the project team to see whether the required analysis and outputs are feasible. The results will be reported in September 2000.

The experiences gained in developing the instruments for these field tests have highlighted many areas where improvements can be made in future projects.

2. The LSID project background

The objective of the LSID project is to determine an appropriate methodology for a longitudinal panel survey on income dynamics. This is an ambitious task with many different facets including:

- reaching agreement on the survey content with the clients
- developing a sample design
- developing CAI instruments and associated field collection systems
- field testing
- specifying systems for estimation, imputation and sample error calculation
- creating and testing an output database

There is little information currently available in New Zealand on income, labour market and family dynamics. Some of the benefits of obtaining this information are seen as:

- understanding what prevents people reaching desirable social and economic outcomes
- assessing the role of policy interventions in creating or alleviating any blocks
- improving the targeting of Government services to those most in need

Achieving these benefits requires information, allowing comparison of the positions of individuals and their families over time and also in relation to particular social and economic benchmarks. The outputs from a survey of income dynamics would be used for developing policies on:

- taxation
- retirement provisions
- assisting people move from income support to paid work
- assisting people improve their income levels
- support for families and children
- education

Statistics New Zealand has obtained additional funding to do this research. Not all of the systems required for a production survey are being fully developed. Instead some aspects are only being investigated and documented to a stage where they are ready to be specified. This provides more opportunity to evaluate alternative approaches rather than just focusing on the construction of production systems.

The project commenced in July 1997 and the first of two waves of field test interviewing took place in July 1999. The respondents were re-interviewed, along with any new household members, in a second field test held in March 2000.

A final report recommending a methodology for a longitudinal survey of income dynamics is due in September 2000. Following the completion of the project, it is hoped that a full survey will be approved. There is a great deal of interest in the research community in such a longitudinal survey as well as wide support across the public sector.

3. The LSID project team

The LSID Project Manager co-ordinated a team of up to 15 people. They have consulted with clients, developed the topic specifications, evaluated hardware, put together training manuals, organised the field work and managed the budgets. The sample design, questionnaire design and instrument programming was undertaken by specialist sections within Statistics New Zealand.

The Questionnaire Design Consultancy section develops questionnaires used for most of Statistics New Zealand 's surveys. The LSID survey instruments are the first they have developed without an associated paper form. Previously paper forms have been created along with new Blaise instruments. The questionnaire design staff involved had no prior experience with Blaise.

Developing Blaise systems is the responsibility of the Technology Application Services division. Over the past three years they have built up considerable Blaise expertise. A full time programmer was assigned to the instrument development. A part time analyst / team leader and several part time programmers worked on other aspects such as derivations and the output database.

4. The LSID design

The LSID aims to collect information on:

- the level of earnings from employment, government support and other income sources over time
- the length of time spent at different income levels
- changes in family status and the length of time spent in different family situations
- the length of time spent in employment, unemployment, part-time work, out of the workforce
- the factors that may be associated with these changes such as education qualifications, health status, age of children, occupation, hours worked, ethnic group and age.

These objectives were developed from the information needs of the clients. Topic specifications were written to cover all the objectives. The topic specifications, detail the different types of data required and were the starting point for the design of the survey questions. In the final report all the topics will be assessed for their feasibility and cost effectiveness.

For longitudinal data the unit of analysis is the individual. Information about each person's family and household is also required. For example, individuals will have a family type, a family income, a household income etc. All LSID data has one of three time-based attributes:

- spell data
- annual data
- point in time data

Spell data relates to a period of time. For example the receipt of government income support payments from start date to end date. Annual data comprises of one value for a 12 month period, such as the income from interest in the 12 months prior to the interview month. Point in time data relates to situations as at the interview date. For example a persons educational qualifications at the interview date.

The LSID is an interviewer administered survey. The large number of detailed questions involved, make it unsuitable for other collection methods. CAI was chosen over pen and paper interviewing because it automates the complex routing, and it allows information from previous interviews to be more easily used.

The field test sample comprised 436 households. Every eligible person, 15 years and over, in a sampled household is asked a personal questionnaire and those under 15, a child questionnaire. The sample size of a full longitudinal survey is proposed to be around 10,000 households.

Other considerations in the design of the survey included:

- how to achieve better estimates for the Maori population through over sampling
- obtaining background information on the respondent's current situation
- calculation of derived variables such as annual income and family income spells
- cross sectional outputs from each survey wave
- treatment of non response, proxy and partial responses
- continuous or windowed interviewing
- the reference period used for data collection
- interviewer training requirements
- strategies for approaching respondents
- incentives to retain respondents
- tracking methods to reduce attrition
- privacy implications
- respondent burden

Longitudinal surveys run by other Statistical Agencies were investigated and the information obtained has helped improve our designs. The Australian Bureau of Statistics carried out two formal peer reviews of the project as it has progressed. Inevitably though there are many aspects that can only be learnt from experience.

5. Development challenges

5.1 Specifying the questions

The questions for the two field tests were developed from the detailed topic specifications. The LSID instruments are far more complicated than any conventional paper based questionnaire and cannot easily be described in traditional ways. Building a Blaise instrument without a corresponding paper form meant a new approach to specifying the questionnaire was needed.

Flowcharts were agreed as the best means of specifying the questions and routing. A very basic flowcharting package, 'Easyflow' was obtained. The designers found this package easy to use and really helpful. It enabled them to concentrate on the job of designing and specifying the questions.

Flowcharts visually depict the flow of the questions involved and document the question texts. They convey the very complicated routing involved in a way that is easily comprehended by everyone involved. Flowcharts were also used as simple paper questionnaires for early cognitive testing. However some things such as table structures are more difficult to show on a flowchart. Over 100 pages of flow charts were produced.

The programmers took full responsibility for writing the Blaise code. The division of labour was simply accepted at the start. By the end of the development of the instruments for the first test this approach was beginning to be questioned. However the pressure to meet deadlines left little scope for changing roles. Originally it was envisaged that the questionnaire designers would be able to access and edit the Blaise scripts but this was never implemented. Only now after the completion of the LSID instrument development is the idea of the question designers working directly with Blaise being re-addressed.

5.2 The development process

When development of the questionnaire started the detailed topic specifications were not ready. The designers began by trying to second guess what was needed. Unfortunately this initial work was not correct but it helped the designers understand the differences between Blaise and paper questionnaires. For example they learnt that it is not possible to ask a question in Blaise with both numeric and written response categories.

The development staff worked together to figure out what was possible. In some places the designers thought it would be better to leave the flowcharts vague so the Blaise programmer could work out the best approach. The programmer suggested alternatives where he thought things could be improved. Most times this worked really well and helped to make the instrument more straightforward. Other times the designers had to make lots of changes because the questions didn't flow well.

Initially the programmer's role was more like a questionnaire designer. He reviewed early drafts of questions made suggestions and comments on how things could be done. When both parties were happy with the flowcharts the programmer would develop the Blaise code. The designers did not really have much idea of how their questions would look like on screen at the start. Once a module was working in Blaise, the designers would review and check it. Then the whole cycle would start over.

The question designers found it a real plus that the programmer was very approachable and good at seeing issues from the questionnaire design point of view as well as a programming point of view. The relationship between the two groups worked well and hopefully this can continue in future developments. There were a few difficulties when staff moved on. New relationships had to be fostered and the protocols re-established.

A lot of work was involved in generating and keeping flowcharts and Blaise scripts up to date and synchronised. Because all the information had to be manually transferred from the flowcharts into the Blaise scripts there was a lot of duplication of effort. And when the inevitable changes were required, the work had to be done over again. In hindsight more training, better procedures and scheduling would have avoided much ‘on the job’ learning and duplication.

5.3 Establishing the timeline

Each set of spell data questions is dependent on the respondent's activities during the reference period. To find out what the respondent was doing they are asked questions about their activities along with the start and finish dates. For example: 'Were you in paid work at any time between date x and date y?'

In Blaise these questions were implemented as a simple table with activity type, start date, end date fields. The activities were then organised into to a timeline. This can be visualised as a sequence showing what a respondent was doing on any given date during the reference period, usually the previous year.

The challenge is not so much recording a respondent's activities but with the many edits that need to be applied. These checks involve finding overlaps and gaps in the timeline. They can only be applied after all the activities are recorded and sorted, otherwise error messages would appear straight away. Getting this to work in Blaise was harder than it might seem. To see whether the interviewer is ready to check the timeline a (Auxfield) question is asked. The field is used to trigger the timeline sorting and edit checks. When all the checks are resolved the field is reset. This means an interviewer can go back and change the timeline without invoking the edit checks until they are needed again.

In the second interview additional activities may need to be included in the timeline. The interviewer first confirms the activities recorded in the first interview then enters any new activities since the first interview into another table. These two timelines are combined using a temporary table and then moved back into the standard timeline. Once the time line is established it determines the relevant spell data questions to be covered.

5.4 Structuring the instrument

The household instrument for the first test had over 700 fields and the personal instrument over 6000. For the second test the household instrument was the same size but the personal instrument expanded to 7800 fields with the addition of questions on the value and type of assets and debt held by the respondent.

When the total size of the LSID instrument became evident it was decided to split the Blaise code into separate household and personal programs. This was a straightforward change as there was little need to share or edit data between the different household members. A separate personal instrument helps ensure that response times do not deteriorate when the number of questions increases. The household and person instruments are linked together for the interviewers with a simple Maniplus interface.

The only performance issue mentioned by the interviewers was the time it takes for the program to open. Some resorted to starting the questionnaire before entering the respondent's house. Once the interview is started, moving between questions is nearly instantaneous. Interviewers were equipped with new Toshiba Portege 3010CT computers running Windows NT Workstation and Blaise 4 Windows. These are lightweight 226 MHz Intel Pentium machines with 64 MB of memory.

The detailed topic specifications were drip feed to the questionnaire designers. The questionnaire specifications were then delivered to the programmer in a similar manner. This meant the over all structure of the instrument was not evident and there was little opportunity to plan or optimise it. Having all the topic specifications to start with would have allowed a better overview and more logical arrangement of the components.

The Blaise data structures for large questionnaires end up being extremely complicated. In some places the programmer tried to simplify the data structure by combining questions asking for the same information. Appropriate question texts for the different situations were computed and one field used to store the answers. While this sounds attractive it makes the route logic much more complicated. This approach will not be used again, as it became too difficult to maintain. Instead data, from separate fields for each question, will be combined with Manipula.

It took a lot of time and effort to get meaningful data from the first pilot test. Cameleon was used to create the scripts to define the Sybase database output tables. Apart from splitting large arrays with Manipula, the Blaise data was exported in its original structure and loaded to the database tables. SAS was used against the database to transform data and derive new variables.

For the second test Manipula was used to export the data in a 'record per question' structure. The data was read directly by the SAS derivation programs and then loaded into the database. While this method required more Manipula code, it simplified the overall process.

Data collected in the first interview is used in the follow-up interview. If the paths to the external files containing the data from the first interview are not specified correctly, Blaise cannot find the data. During development the instrument was run on our internal network. This required different external file path definitions than were eventually used on the interviewers' laptops. Mapping the path to a drive letter, such as E: did not always work. To fix the problem the full path to each external file had to be specified in all cases. This of course is not ideal, as the Blaise scripts have to be altered and re-prepared for each environment.

5.5 Managing changes

In any large development it is inevitable the requirements will evolve and be refined. How these changes are managed is often the difference between a successful or a stressful development. As milestones approach, and it becomes obvious that there is more work than can be done in the timeframe left, priorities have to be determined and the hard decisions made about what will be left out. Often the need to accommodate change is not allowed for in the project planning and scheduling.

Late delivery of topic specifications meant questionnaire development also ran behind schedule. Frequent changes to the question texts also became a significant issue. The large number of variables used as fills in the question text along with the poor methods used to identify changes in the flowcharts and transfer them into the Blaise scripts all helped to compound the delays.

Given the designers inexperience with Blaise and the complexities involved it was not considered viable for them to manage changes to the question texts in Blaise. With appropriate training and systems it would be more efficient for the question designers to maintain the texts themselves.

The delays with the topic specifications worked against an orderly sequence of development and also impacted on the time available for specifying, testing and programming the questions. Modules developed in the earlier phases needed to be revised in light of later topic specifications. All the changes required stacked up to the point where they impacted on the ability of the programmer to effectively deal with them in the time available.

5.6 Validating the instrument

Testing a CAI instrument is vital. Considerable time and effort was spent in cognitive testing as well as ensuring the question texts, routing and edits were correct. The need to refer to data collected in the first test considerably increased the complexity of the routing and editing for the second test instrument and thus the amount of validation required.

The question designers carried out cognitive tests using their flowcharts. Feedback on the flow and comprehension was used to shape and refine the questionnaire specifications. However the opportunity to 'desk check' the questionnaire using the flowchart was not utilised as extensively as it could have been. Mostly this was due to time and resource constraints. Testing the route did not really commence until much of the programming was completed. This meant a lot of effort was required over a relatively short period.

The number of ongoing changes also impacted on the timeframes available for testing and validating the instruments. Modules had to be rechecked again and again. The lists of changes became longer and longer as the project went on. As there is always room for improvement it was not until the pressure of impending deadlines became obvious that the issues were prioritised and the modules finalised. It would have been preferable if individual modules were signed off and put aside earlier, so efforts could be focused on higher priorities.

Many of the people who tested the first field test instrument had moved on by the time the second instrument was ready for checking. Training new people in what the instrument was intended to accomplish and helping them distinguish genuine errors, took almost as much effort as the testing itself. A variety of people were involved in the instrument validation, including people outside project. The instruments were checked against the questionnaire specification flowcharts. The fact these specifications were in an easily understood format helped the testers determine whether the behaviour of the Blaise program matched the documented requirements.

The majority of the routing involved in the LSID instruments was programmed using 'State Rules' technique(1). Although this method requires a lot more lines of code than the usual Blaise rules section, it makes the route logic much easier to implement and validate. The routing conditions for each question can be written and verified independently of surrounding questions. This is especially beneficial when changes or corrections are needed, as testing can be confidently focused on only the altered questions.

Incremental validation would certainly have spread out the huge workload involved in checking the instrument. If individual modules were programmed, tested and signed off as the development progressed then the final 'build' would simply involve checking that everything had been assembled correctly.

6. Interviewer training and support

Interviewer training for the first field test included pre-course reading material, followed by a five day face to face training session, covering laptop use and specific LSID survey content. Nineteen interviewers attended. The project team developed the training course with assistance from Statistics New Zealand 's Training and Development section.

The interviewer manuals and support materials were very well received. After the five day training, and a week of practice prior to going into the field, the majority of interviewers were confident about their ability to conduct an interview. The main changes made for second field test training were to keep training groups smaller (no more than 8 people) and to have as much hands on practice as possible. Interviewers requested more detailed information regarding the survey content prior to the training course.

A 0800 (free phone) number provided first line support for interviewers and respondents with problems, queries or concerns. Subject matter experts provided further assistance as required. Most calls from respondents were general queries about the survey. In total over 100 calls were received for both tests. The interviewers found the free phone number helped to resolve problems and provide guidance quickly and effectively especially during the initial settling in period.

7. Field test results

The first field test, held in July 1999, aimed to:

- provide an indicative response rate
- trial the CAI instrument and proposed field procedures
- trial performance of laptops and accessories
- assess respondent load

Overall the Blaise instrument worked successfully. Only a couple of minor routing errors were discovered. Interviewer reaction to the laptops and the Blaise system was very positive, with almost unanimous agreement at the debriefings that more questionnaires should be done this way. Feedback from interviewers and analysis of results so far indicate respondents had few difficulties with the questionnaire content.

The laptops proved reliable with only a couple of problems very early on. The screen quality of the laptops was excellent. However there were the usual problems when interviewing in strong light. Some interviewers accidentally hit the F2 key (which exits the questionnaire), while using the numeric 2 key. This problem was resolved by reassigning the F2 key function. Two batteries provided ample power for a day's workload and respondents had no objections to interviewers plugging in on the odd occasions when it was necessary.

The average interview length for the personal questionnaire was around 30 minutes. This is well within the upper limit target of 45 minutes on average. Interviewers indicated the length of the personal interview varied greatly, depending on the complexity of the household and each individual's situation. Interviewers commented that respondent fatigue was a problem with the longer interviews and stressed the importance of preparing respondents to help avoid difficulties.

Over 70% of eligible individuals provided a full response. Item refusals were very low overall with only very small numbers of respondents refusing to answer income questions. Offering the choice of answering to the nearest $100, $1000, and finally income ranges seems to have been very successful in minimising refusals and don't know responses to these questions.

Respondents were asked to provide a contact person to assist with follow-up and 21% answered "no" to this question. While this appears high, interviewer feedback indicated that some of these people felt that the information was not required, as they had no intention of moving, while others may have had difficulty providing a contact. The real impact of this non-response will not be known until we analyse the contact results from the second field test.

Each interviewer's work was pre-loaded onto their laptop. Interviewers transferred their completed cases onto floppy disks and mailed them to Statistics New Zealand each week. This eliminated the need to develop a full-scale case management system. Wrapping disks in plastic bubble wrap and using courier bags provided ample protection at an acceptable cost. Of the 80 disks returned only one was damaged, most likely before posting.

The data was amalgamated for in-house processing such as coding of occupation, industry and education qualifications. Analysis of the first field test cases helped identify potential problems with the questionnaire and will contribute to developing cost estimates for the full-scale survey.

The second field test was conducted in March 2000. This test involved the adults who responded to the first test, together with anyone else living in the household at the time of the second interview. The main aims of the second test were to:

- provide indicative overall panel and longitudinal response rates
- trial the asset questions
- assess the respondent load
- trial interviewing using data from the first wave of interviews
- further trial performance of laptops and accessories and proposed field procedures
- provide data for testing database design
- provide more information to estimate costs for a full survey

8. Lessons learnt from the LSID development

The experiences gained in developing the instruments for these field tests have highlighted many areas where improvements can be made. Some changes can be easily implemented others will take longer and require more research. Key considerations include fostering better communication between all the parties involved in the development more effective and more disciplined development methods and improved project management.

The question designers clearly need to understand what it is possible to do with Blaise. The challenge involved in creating such a complex questionnaire was more than enough for them to deal with at the start. Learning to write Blaise code at that stage may have been counter productive. Now there is a much keener appreciation of the benefits of questionnaire designers developing their own Blaise programs. A recent Blaise training course for the designers was enthusiastically received. Understanding what is involved in converting flowcharts into a Blaise program will at the very least improve the communication with the programmers.

The programmers and questionnaire designers could benefit from a working environment more conducive to project work. There is a need for places where all the developers can meet to discuss issues, and also quiet areas where they can concentrate on writing flowcharts or code. Another idea is to have at least two Blaise developers in the team. This would allow one to be the main contact and be available to consult, propose alternatives, to understand the topic specifications and learn about questionnaire design. The other programmer would also be involved but able to concentrate on writing code as well. The main developer could be shared amongst several projects and act as a 'mentor' for less experienced programmers. This arrangement also ensures there are knowledgeable backup staff available.

Flowcharts have proved to be an excellent communication and documentation tool. They successfully moved the designers out of the mode of creating paper questionnaires and provided a common reference point understood by both technical and non-technical staff. There is scope now to work towards refining the way in which flowcharts are used. Current flowcharts only represent the interview, and do not include all the programming elements. Ways to represent some design aspects, such as tables, need to be worked out. Other items, such as the field identifiers and types need to be included. This information would help with generating code, testing edits, creating derivations and with resolving interviewer problems. Spell checking and editing to get the question texts accurate and finalised earlier is another area where attention is required. More sophisticated flowcharting packages are being evaluated. These may allow additional fields to be specified and extracted.

For large projects formal methods of documentation, change control and program version control are necessary. Informal communication can mean systematic and or documented outcomes are not generated. For example, written amendments are preferable to asking for changes via phone calls. The challenges for those who will have to maintain the instrument in the future need to be considered. Implementing a more formal change control process to keep track of questionnaire amendments and any consequential impacts on other questions is a priority.

A repository, where question specifications can be documented and re-used, is an attractive proposition. Initially a simple Lotus Notes database was set-up. This was not used due to the overheads of capturing all the specifications and a perception that it did not provide any utility. For a repository to be useful has to be kept up to date with any changes. Ideally it would be linked to the flowchart specifications and allow the extraction of information for use by other programs. Providing different users with views of the specifications tailored to their needs is likely to be an incentive for them to put in the required effort.

Using a repository could also facilitate better change control processes. If designers specified and maintained their question texts in such a database it would hopefully lead to greater efficiency and consistency. Various staff could contribute different aspects of the specification. For example the Blaise programmer might determine the field identifiers, while the question designers define the texts and field types. Better procedures for program version control and for carrying out testing could also be supported with a repository. Linking flowcharts with a specification repository would eliminate duplication and save a considerable amount of work. How flowcharting tools can be integrated with a repository is being investigated.

Cameleon's potential to access and manipulate Blaise metadata was not fully exploited. New uses for the Blaise metadata could be developed especially when the Blaise language is understood and used by all the designers. Setting up scripts to create database tables for the LSID is only a small example of how Cameleon could be employed.

The instrument development sequence used in this project, where flowcharts were prepared and then programmed in Blaise, is somewhat cumbersome and caused delays. Other approaches are being considered. For example the question designers could write their own prototype Blaise code. The programmers could then focus on integrating modules and look after quality management issues. With appropriate training and support it should be quite straightforward for questionnaire designers to set-up and maintain simple prototype questionnaires themselves. Team members could then carry out cognitive testing using Blaise and review questions at a much earlier stage.

Having the ability to quickly document, build and demonstrate questionnaire proposals would have helped make the LSID development more effective. Running programs on a laptop as early as possible also ensures there is time to understand and address any issues associated with the laptop environment. It could be useful to include some fields in these prototype questionnaires to document feedback on the proposed questions. Finalising questionnaire design earlier will reduce rework and the overheads associated with making changes after the whole instrument is constructed.

The facility in Blaise to add descriptive information within the field definition caught the attention of the questionnaire developers during their training course. They were quick to see the potential to record information relating to the question development process, such as when changes were made and by whom. Cameleon could be used to extract and manipulate these descriptions.

The development process involves refining the detailed topic specifications into a questionnaire. Establishing a high level flowchart at the start would give all the team a sense of the whole questionnaire much earlier in the development process. Clarifying the big picture first and then working into the detail later provides a sense of direction and ensures the overall structure is considered.

A corresponding high level Blaise program structure is also needed. Now that the development of large-scale electronic questionnaires is an ongoing activity in Statistics New Zealand, the impetus (and funding) to standardise and reuse existing code has increased. Many of our household surveys have a common high level structure and a standard Blaise 'template' is planned to formalise this. The template instrument would include the standard survey management questions such as the scope and coverage rules etc. Survey specific modules could then simply be added as required.

All the developers need to work together to initially plan the high level structure of the survey. Deciding which related groups of questions fit together into modules a fairly early stage would help with organising and scheduling the development work. Modules could then be developed and tested as they are detailed.

A modular building block approach implies that related sets of questions, such as income or house ownership, are specified, written and tested discretely. This contrasts with the LSID development where large parts of the questionnaire were specified, before any programming started. Edits were added even later and testing only commenced once the whole instrument was completed. This process meant there was less opportunity to revise and correct the instruments once they were built.

Edits, and any associated questions to assist in resolving edit failures, need to be designed in from the beginning instead of at the end of the development sequence. Adding new questions and edits or changing the sequence or routing of questions often involves consequential changes. All of these things naturally increase the amount of rechecking required and contribute to the long development times. Developing and testing edits earlier in the development cycle would reduce the number of last minute changes.

Many of the scheduling and change control difficulties are typical project management issues. Managing deadlines, prioritising and making trade-offs were tough decisions especially when they involved compromises that made it difficult to bring the programs into line with what was wanted. Decisions often took a long time to be made.

Ensuring target dates are met helps to minimise consequential impacts. The lack of previous benchmarks made it difficult to estimate the time it would take to develop and test the instruments. More time and

resources were needed especially for the cognitive and usability testing. It would also be beneficial to have cognitive testing completed before serious programming is undertaken. Quicker turnaround of changes along with a formal confirmation of programming interpretation before module testing will reduce some of the larger development iterations. Given the benefit of our experiences more realistic timeframes to complete validation can be scheduled in future. Combined with better planning and training, more systematic overall validation will be possible as well.

9 Conclusions

Overall this project has been a success. The skills of the LSID team members along with their ability to work together and deal with the stresses involved has contributed greatly to the positive outcome. Everyone focused on what they could improve, rather than looking at what others might do better.

Blaise 4 Windows performed superbly. It met all of the challenges that come with such a large and complex interview. Very few Blaise programming issues arose during the development and the instruments worked flawlessly in the field.

The two field tests involved in the LSID research have been extremely valuable in helping build our capacity to develop Blaise instruments. Managers have come to understand and appreciate the requirements and benefits of CAPI in a very tangible way, without having to deal with the risks usually associated with implementing production CAI surveys. These experiences have favourably influenced the direction of CAI within Statistics New Zealand. Funding and resources have recently been allocated to implement two production CAI surveys during the 2000/2001 financial year.

In any development, complexity initially arises through the tendency to make use of all the available options. This seems to be an inevitable part of learning. With experience it becomes easier to recognise what can be left out and over time adopt simpler approaches. New systems and better processes will also assist.

Everybody involved has learnt a lot. While some have commented this project 'has taken years off their life' generally there has been a high level of personal satisfaction with what has been achieved. A number of staff who worked on this project, have now rotated into other CAI developments where their experience will be of much benefit. The knowledge gained over the past couple of years will prove to be very valuable for future developments.

(1) IBUG Newsletter March 1995 'Blaising the Paper Route - A method for converting paper questionnaires to Blaise ' describes the 'State Rules' technique.