Providing Computerised Support For The Ad Hoc Use Of External Statistical Data By Casual And Novice Users

Peter N. Hyland
Joseph G. Davis

Department of Business Systems
University of Wollongong

Abstract

This paper describes the development and testing of a system that provides casual and novice users with simple yet powerful tools to manipulate statistical data from external sources. The prototype combines Multi-Dimensional Databases and Online Analytical Processing tools to produce a generic Web-based interface compatible with a wide variety of types of statistical data. The prototype has performed successfully with complex, high-volume statistical data typical of that in general use. It has been positively evaluated by casual and novice users and is now being used to further explore the latent and explicit needs and abilities of these users.

Keywords

Statistical systems; OLAP; System evaluation

INTRODUCTION

This paper concerns the problems that arise from the ad hoc usage of external statistical data by casual and novice users. External statistical data (ESD) is statistical data intended for use outside the organisation in which the data originates. As such, ESD is used by a wide group of users including many casual and novice users such as business managers, researchers, students etc. These users face a number of distinctive problems. Existing tools are often hard to learn, vary greatly in the way they present data and offer too little functionality for ad hoc analyses. Moreover, ESD is multi-dimensional and any set of ESD may contain too many or too few dimensions that a user needs. Similarly, the level of aggregation is often unsuitable, being either too detailed or not detailed enough. Existing tools do not adequately address the problems associated with dimensionality and aggregation of ESD. Many of these problems arise from a lack of understanding of casual and novice users and of ad hoc statistical analysis.

The research demonstrates one approach to solving these problems by integrating multi-dimensional databases (MDDB), on line analytical processing (OLAP) and the World Wide Web. These technologies have not been used with this type of data or with this type of user. The design and development of the prototype system, *Abacus*, have indicated the suitability of the approach while reflective observation of the design process has given rise to guidelines for the development of similar systems. The critical evaluation of *Abacus* that is now under way, will extend our knowledge of casual users and enable further refinement of the prototype.

In the following sections, we will:

- define the main entities in the research,
- suggest a solution to the problems identified,

- present the research method to be used,
- describe a prototype implementation of the proposed solution,
- propose a general method for the development of similar systems in a variety of contexts,
- outline the evaluation of the implemented prototype.

THE RESEARCH PROBLEM

The theoretical framework for this study is an adaptation of a widely used framework for the study of Information Systems and is shown in Figure 1 (Leavitt, 1965; Jenkins, 1982).

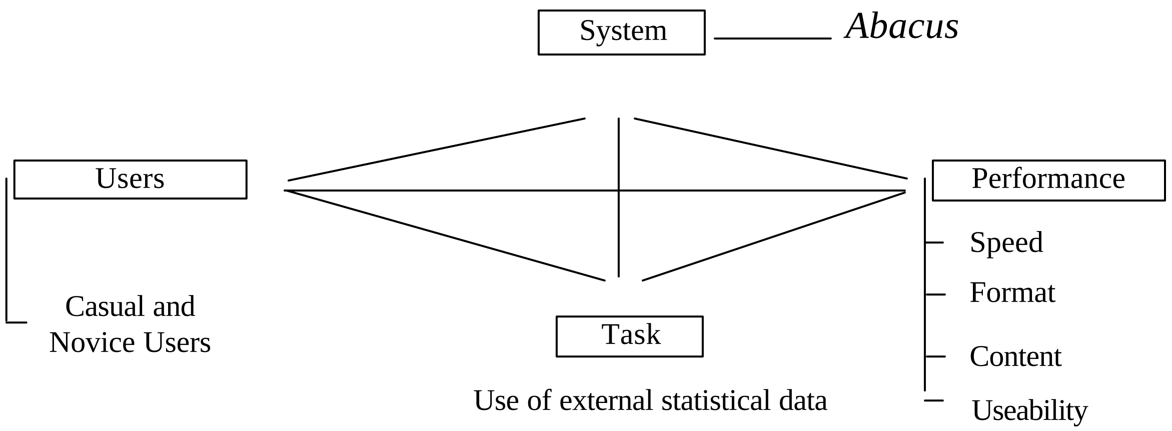

Figure 1 Research Framework for the study

To understand the problem, one must understand the following elements of the framework.

Nature of External Statistical Data (ESD)

There are four significant features of ESD that need to be clarified.

- Statistical Data. The term "statistical data" has been described by authors such as Rafanelli (1989) and Ozsoyoglu & Ozsoyoglu (1985) as either individual records from a data gathering process or the data resulting from the summation or counting of such records. This description encompasses many types of data including population data, trade figures, economic indicators, health statistics etc.
- Source. ESD is statistical data provided by a source organisation to persons or organisations outside the source organisation. Typical sources of ESD are government departments, national statistical authorities, international agencies and so on.
- Level of aggregation. The level of aggregation of a set of statistical data is the extent to which the data has been summarised from the raw data. The lowest level of data available is called unit data e.g. in census data, the unit data are the responses from each individual. Data can be aggregated at many levels such as the population in each state or in each town.

  For privacy or security reasons, complete sets of unit data are seldom provided to external users. Instead, ESD is usually provided as:
  - samples of unit data.
  - complete sets of unit data with small numbers of artificial members added.
  - standard profiles i.e. data that has been aggregated one or more levels above unit data.

  All of these formats are used by providers of ESD, often resulting in unsuitable or inappropriate levels of aggregation for ad hoc users whose focus is subject to change.
- Dimensionality. A dimension can be any relationship between items in the data that is of interest

to the user and can be used as an aggregation path. In a census, for example, dimensions could include age, ethnicity, income and so on. These terms describe relationships that exist within the data and form a suitable aggregation path.

Within any dimension, there are a number of possible states called “members”. For example, sex has only basic 2 members, male and female. The number of members in different dimensions varies greatly. Sex has only 2 members and few levels of aggregation while geographical location has many members and can be aggregated in many ways e.g. into regions, states, countries etc.

The Task Carried Out by Casual and Novice Users of ESD

To get the data that they need, it appears that users of ESD typically carry out the following:

1. Identification of a data set containing all necessary dimensions,
2. Selection of the dimensions required in the analysis,
3. Organisation of the dimensions in a meaningful way,
4. Aggregation of the data to the required level,
5. Formatting the output

Collectively, these activities are described as "analysis" in the OLAP literature and the term will be used in that sense in this research. It should not be confused with statistical analysis such as the production of correlation coefficients and other statistical coefficients.

The Nature of Casual and Novice Users

Although there have been many attempts to classify users of information systems according to their level of experience, depth of understanding, frequency of use and so on, these classifications are not widely agreed upon and are ambiguous. To overcome this ambiguity, both Cuff (1980) and Fisher (1991) abandon the attempt to <u>define</u> user types and recommend that researchers identify the characteristics that apply to the type of user being studied and describe the users in terms of these characteristics. This approach will be used in this research.

The characteristics most often associated with casual and novice users are that they have :

- a low frequency of use (Bannon, 1995; Shneiderman, 1995; Preece, 1994; Codd ,1974)
- a high variability of tasks done (Eason, 1975; Lough and Burns, 1977; Codd, 1974)
- a short duration of use (Fenichel, 1979)

Table 1 Categories and cut off points for user characteristics.

Characteristic	Categories	Boundary values
Frequency of use	Infrequent	less than 4 per month
	Moderately frequent	4 to 15 per month
	Frequent	more than 15 per month
Duration of Use	Short duration	less than 2 years
	Moderate duration	2 to 5 years
	Long duration	more than 5 years.
Task Variability	Low Variability	less than 3
	High Variability	more than 3

In this research., a casual user is one who uses a system for a variety of tasks and does so either with low frequency over a short or moderately short period of time, or a moderately frequent user who has used the system for only a short period of time. The group represented by this definition comprises a significant proportion of the total user population and is one that has particular problems with the use of ESD because of the characteristics of the group itself.

Features of the Casual User

Cuff (1980) suggests that casual users share important features in common and that some of their system requirements can be inferred from these features. The following list of features are representative of those found in the literature. A casual user might exhibit:

- poor retention of major and minor details of how the system works
- a propensity for errors and the need for a safety net
- acceptance of only a brief period of training
- intolerance of formality in an interface and strict formal query language
- poor handling of the logical construction of queries
- little knowledge of database details
- a preference for a range of query precision including browsing

The list of features helps define the system interactions that need to be provided for casual users and many of these will be incorporated in the user requirements in the following section.

REQUIREMENTS FOR A POSSIBLE SOLUTION

In the previous section we have discussed the nature of ESD, the task of analysing of ESD and the characteristics and features of casual users. Based on this discussion, and pilot studies conducted in this research (Hyland and Hasan, 1997), we now propose a set of system requirements for a system that would support casual users of ESD in a wide variety of settings

The system should be able to meet as many as possible of the following. It should:

- be easy to learn in less than half an hour (Cuff, 1980; Case, 1986; Eason 1975)
- require the user to remember very little about system operation i.e. be highly intuitive.
- provide a model of the available data (Burgess and Swigger, 1986)
- provide a flexible and interactive interface.
- be able to access data from a number of sources. This would include the ability to
  - store and access data from a number of sources with different storage formats
  - provide a seamless interface between data from different sources
- be able to find the number of individuals in a population with a set of characteristics
- be able to produce 2, 3 and 4 way cross tabulations in a short period of time
- allow the user to manipulate the position of row and column variables in tables
- allow the user to view data at different levels of aggregation
- allow the user to view subsets of the data or to filter out unwanted variables
- provide a simple means of integrating data from the system into reports and documents
- be accessible from personal computers at the workplace and at home.

Combinations of several recently developed technologies have the potential to meet the vast majority of these system requirements. They could provide extremely flexible solutions for casual users in any workplace and with any type of ESD e.g. census data, health data etc.

A Strategy for Satisfying the Requirements

Requirements such as having a flexible, interactive interface or being easy to learn and easy to remember, strongly suggest the use of a Graphical User Interface (GUI) but are not very specific as to functionality. However, the ability to provide 2, 3 and 4 way tables, to control the level of aggregation and to view subsets of the data suggests the use of OLAP. The multi-dimensional nature of ESD, the need for an underlying model of the data and the need to be able to access different data sets in a common format all suggest the use of MDDBs.

OLAP has been designed to carry out complex, multi-dimensional analyses of business data. It provides functions for tabulating data, filtering out values, slicing through the data to view a particular sub-population and drilling up and down to different levels of aggregation. MDDBs provide a method of defining models of large data sets and storing data from a wide variety of sources. There is no apparent limitation to the types of statistical data that they can hold. (Pendse, 1999; OLAP council, 1997; Codd and Salley, 1993).

Unfortunately, OLAP/MDDB systems are usually expensive and produce very large files that would make it difficult to provide access to casual users outside of the source organisation. However, the outputs from OLAP are often quite small tables that could easily be transferred via the World Wide Web. Many OLAP/MDDB systems have recently produced Web versions but little or no research has been carried out on their effectiveness.

Based on the functional requirements above, the combination of OLAP, MDDB and the Web appear to provide one approach to meeting the needs of casual users of ESD.

RESEARCH METHOD

The research reported in this paper has three distinct but closely related goals:

- To demonstrate that the proposed approach is feasible and appropriate for casual users by developing a prototype system,
- To understand the process of building systems for the ad hoc use of ESD by casual users.
- To learn more about casual users through observation of their use of the prototype

All of these goals can be accommodated by a single, very general research methodology - the use of system development as research i.e. the development of a theoretical solution to a real problem and the construction of working prototypes that employ that solution to demonstrate the viability of the system in everyday use (Weiser, 1993). This methodology is widely used in experimental computer science (Wynekoop and Conger (1991). To address all of the goals above, a number of research methods are needed. System development as research typically follows a multi-methodological approach and so is particularly suited to this research. (Nunamaker et al, 1991; Vogel and Nunamaker, 1990). This research will use a "system development" approach centred on the design, development and evaluation of a prototype.

THE *ABACUS* PROTOTYPE.

The development of the prototype required a set of tools that would support the production of MDDBs and the development of Web-based, OLAP applications. The prototype, *Abacus*, has been built using Gentia, a client/server, OLAP/MDDB product. The prototype was developed on a Pentium personal computer running Windows NT. Both the applications and data sets that comprise the prototype have been stored on a Unix server. The browser used in testing was Netscape Communicator v4.5.

A set of complex data was needed to explore the process of developing the MDDBs. This data should be typical of typical of ESD in general use, having many records, many dimensions and some dimensions with many members that could be organised into multiple hierarchies.

The data selected for this project is a 1% sample of the 1991 census of Australia. The data contains the unit data for each of 63000 dwellings, 67000 families and 168000 people. There are over 60 dimensions and the number of members in each dimension varies from 2 (sexes) to 58 (field of highest qualification). Although the data has fewer records than was originally planned the data was still considered to be both large and complex enough for test purposes.

Development of *Abacus*

The first goal of the research is to demonstrate that the proposed approach is feasible and appropriate for casual users. To satisfy this goal it is necessary to do the following tasks:

- produce a number of MDDBs containing typical ESD
- produce a common OLAP interface to these MDDBs
- demonstrate that these meet the system requirements listed above.

Production of the MDDBs

One requirement of the prototype is that it operates with different types of data. Ideally, these data should be from different domains but this would have made the evaluation of the prototype by users too complex. To overcome these contradictory requirements, the census sample file was loaded into three separate MDDBs, one for each of Dwellings, Families and People. Although the data in these MDDbs are related, the use of the interface with several types of data could still be demonstrated. This reduces the complexity of each MDDB but the resulting smaller MDDBs are still quite complex. All of the MDDBs produced have multiple dimensions and within each of the dimensions there are one or two with a large number of members and one or two of the dimensions were structured as multi-level hierarchies.

MDDBs store the definition of the data that they contain i.e. the number of dimensions in the MDDB, the members of each dimension and the hierarchical structure of each dimension. Gentia provides two types of MDDB. The first, Gentia Analytical Data Bases (GADBs) store both the definition of the model and the actual data. The second divides the task, storing the definition of the model in a Business Model and the data itself in a Base Model.

Once a model has been defined for each GADB, data can be loaded into the GADB by running a program written in a proprietary language. Initially, GADBs were produced from the People data using only three or four of the available dimensions. These were loaded with data directly from a comma delimited text file and resulted in relatively small data files.

To see if the GADBs were functioning properly, a simple application was built using the Gentia application interface. The tables produced by this application were checked against source data and were found to be correct. Larger GADBs containing more dimensions and hence more members, were then produced but did not function properly, as explained below.

Separate Business Models were defined for the Dwelling, Family and People data and from each of these a number of Base Models were generated. Loading data into the Base Models from the text file involved the creation of a "task" using a GUI and writing some simple code. Small Base Models were generated for each of the three primary data sets.

Because both the GADBs and the Business Models contain the definition of the data, the definitions are available to Gentia applications. With only minor alterations, an application can be modified to use either a GADB or a Business Model correctly. Having made these alterations, Base Models were shown to function correctly, as long as they were small.

MDDBs with many members, whether stored as GADBs or Base Models, took many hours to load and produced huge files. The approximate file size required for Gentia MDDBs is given by the product of the number of members in each of the dimensions in the model. An MDDB containing all of the dimensions for the Family data would be over 314 gigabytes in size.

A Join Model is a Base Model that contains all of the dimensions and members in two or more previously defined Base Models. It does not contain any data, only a model definition. Gentia generates the numerical values to populate the Join Model from the stored values in the component Base Models when the Join Model is used. Join Models perform more slowly than Base Models but make larger models possible without a corresponding increase in file size.

We observed that Join Models did not correctly combine all of the Base Models from which they were formed. The values displayed were correct only for the first Base Model that comprised the Join Model. When a dimension from the second Base Model was included in a table, the values for that dimension were the total for the population. We have been unable to resolve this problem and so it is a major limitation of the Business/Base Model approach.

To overcome the problem of file size, MDDBs containing sub-sets of the three primary data sets were produced. These contained five dimensions each, which was thought to provide sufficient complexity for the user evaluation phase. The sub-sets worked without apparent problem, demonstrating the feasibility of producing several MDDBs containing typical ESD.

Production of a generic OLAP Interface

The application mentioned above was only used to test the accuracy of the MDDBs. The final *Abacus* interface is much more complex. For example, it supports limited browsing by allowing users to select an MDDB and view the dimensions available in that MDDB

The *Abacus* interface provides two modes of interaction, namely the production of:

- a single value e.g. the number of single men over the age of 65 with an income of $50000
- a multi-level table e.g. the numbers of men grouped by age, marital status and income

Value Mode

Interviews with casual users of ESD suggested that users often want a single value. Although such values can be found in a table, it was decided to build an interface that could bypass the need to create a table and which might be simpler to use. This gave rise to a style of interaction called “Value Mode”. Value Mode operates by letting the user define multiple rules and applying these to the whole population. A Value Mode screen is shown in Figure 1.

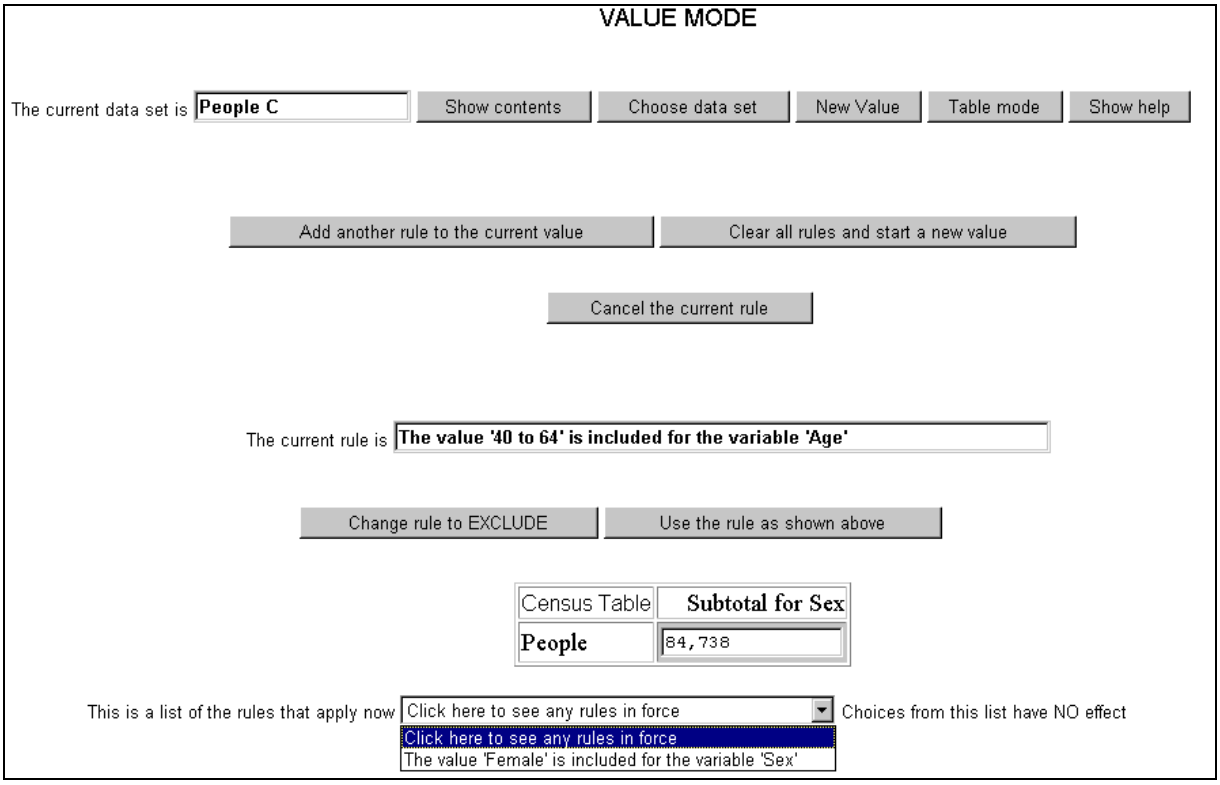

Figure 1 – A screen from "Value Mode" showing the process of defining a rule

When the user selects a button labelled "Define a rule", *Abacus* provides a list of available dimensions and asks the user to select the dimension to which the rule will be applied. *Abacus* then presents the list of members for that dimension and ask the user to select one. Once the member has been selected, a rule of the following form is displayed:

The value " member name" will be included for the variable "dimension name".

An example of a rule is shown in the centre of Figure 1. Once a rule is defined, new buttons are displayed on the screen. These let the user change the sense of the rule from include to exclude, to apply the rule or to cancel the rule. Once a rule is applied, the selected member is filtered from the value being displayed in a one celled table. Filtering is done either positively or negatively, depending on the wording of the rule. The user can apply multiple rules until the desired sub-set of the population has been fully described e.g. all females between the ages of 40 and 64 who speak Greek. A drop down list shows the rules that are currently in force and the headings in the one-celled table indicate the dimensions that have been constrained.

Table Mode

Table Mode, shown in Figure 2 shows, allows the user to create multi-level tables and has been tested with up to 6 dimensions. The buttons in the upper row are navigation functions while the central block of buttons provides the OLAP functions. The first two buttons on the left of the block let the user add or remove a dimension from the table, as either a row or column. The next two buttons let the user move a dimension inwards or outwards on multi- level tables. The lower of the next two buttons start the same "define a rule" process used in Value Mode while the upper one hides or shows the slicer – the slicer is visible in Figure 2, as a set of drop down lists below the OLAP buttons. The next two buttons control processes for moving dimensions from one axis to the other e.g. moving a row to a column.

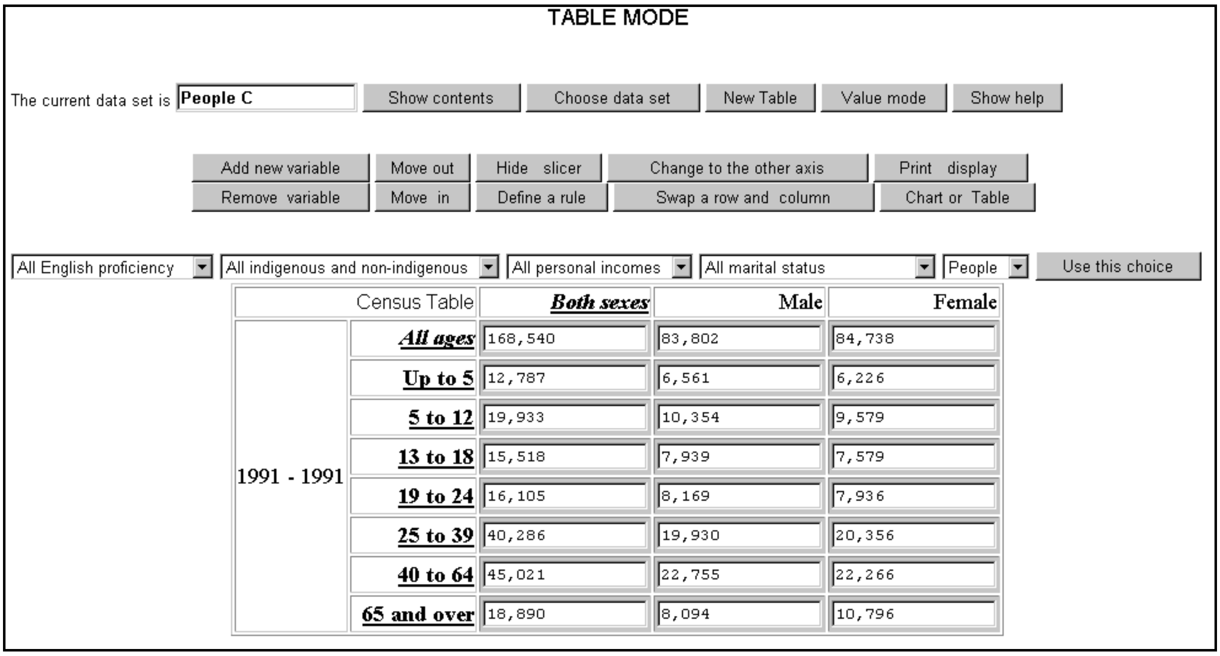

Figure 2 – A screen showing "Table Mode".

The table in Figure 2, is a 2-way table by age and sex. Both the age and sex dimensions have been disaggregated one level (drilled down). In the age dimension, each of the age groupings at this level are underlined to show that they can be disaggregated further while members of sex dimension are shown in plain text to indicate that they cannot be disaggregated further.

EVALUATION OF *ABACUS*.

To complete the first goal of this research, it must be shown that *Abacus* satisfies the system requirements described earlier. To demonstrate that both functional requirements and useability requirements have been met, it is necessary to test the performance of Abacus with a representative sample of casual and novice users. Observational studies of a sample of casual and novice users are being carried out, both to evaluate the prototype and to learn more about the nature of casual users (see Future Research).

Participants in these studies are given about half an hour's training in the use of the prototype and are then videotaped while carrying out a set of representative tasks. The 18 tasks include using Value Mode to find a value with 2, 3 and 4 constraints and using Table Mode to manipulate 2, 3 and 4 way tables. Participants then evaluate the ease of use and efficiency of the prototype using an instrument based on the Technology Acceptance Model (Davis, 1989).

The results from 25 participants who have completed the testing so far are very positive. On average, users get 80% of the 18 tasks correct suggesting that the interface can be used correctly. User evaluations of the ease of use and the efficiency of the prototype are also very positive with the average ratings on two 7 point Likert scales suggesting that the prototype is reasonably acceptable to the target group of casual or novice users.

While the results of user testing suggest that the combination of OLAP/MDDB and the Web do, in fact, provide a viable solution to many of the problems of casual users of ESD, they also indicate some limitations in the performance of the prototype.

Limitations of the *Abacus* prototype

Although the initial testing was largely successful, it revealed a number of limitations in *Abacus,* some of which were mentioned above. Other limitations included:

- The need to use multiple MDDBs to deal with the problem of large file sizes has led to a loss of completeness in the data. *Abacus* can only produce tables from within one MDDB and so is unable to find any value that depends on dimensions from two different MDDBs.
- *Abacus* can produce tables with any 3 dimensions but may not be able to include a fourth or fifth dimension. The volume of data being processed by a calculator function causes this. Removing the calculator would solve the problem but reduce the functionality.
- Gentia retains information about the state of a table if the "Context" setting for that table is turned "On". To produce the sub-totals used in tables Context must be turned "Off", causing certain actions to return the table to its default state e.g. disaggregated dimensions on a table return to their aggregated state when a new dimension is added to the table
- The production or alteration of large tables imposes a significant burden on the Gentia server, resulting in appreciable delays. In client/server mode, these delays are acceptable but when using a Web version of *Abacus,* the performance deteriorates further. Performance would be unacceptably slow on any line speed less than 28Kb/second.
- The Web version of Gentia does not support screen colours or the use of explanatory help when a cursor moves over a button. This means that the labels on all buttons must be self-explanatory and so these labels are quite long. This reduces the options for screen layout.

TOWARDS A GENERAL METHOD FOR BUILDING ESD SYSTEMS

The second goal of this research is to produce a general method for the development of ESD systems using OLAP/MDDB and the Web, based on our experience of developing the *Abacus* prototype. Some initial recommendations can be made about the process as a whole and about the design of the MDDB and the OLAP interface, in particular.

Overall Development Process.

These recommendations are appropriate for the development of systems to support the analysis of statistical data by casual or novice users, either inside or outside an organisation.

First, determine if there are any casual or novice users for whom such a system is needed as even a simple system will be costly. If the users are not casual users, then other systems may be more appropriate. During this process, both the data requirements and the functional requirements of users should be established for use in subsequent steps.

Decide which data to provide to users and provide mechanisms for ensuring privacy/ security. Identify OLAP functions that will be necessary to meet the functional requirements of users, and so determine the minimum functionality of the OLAP development tools to be used.

Use the description of the data that the source is prepared to make available and the minimum functional requirement of the OLAP tools to select both the MDDB and the OLAP development environment. Ensure these are compatible with existing hardware and software.

Having resolved these issues, it is possible to design the MDDBs and the OLAP interface.

Design of the MDDB

The following stages were identified in our development of the prototype:

1. determine the number of entities that are needed, the dimensions for each entity and the members in each dimension. Ensure that there is unit data available to populate these entities.

2. determine the hierarchies for each dimension. These hierarchies will be defined later but the number of new members required by these hierarchies is used in step 3.

3. calculate the size of the files needed to hold all the members, including hierarchical ones, for each of the dimensions in each entity. If there is insufficient storage, partition the data.

4. partition the data if there is insufficient storage space. This requires the database designer to know what combinations of dimensions users require. Any dimensions that are used together must be stored together in an MDDB. Multiple MDDBs will probably be necessary.

5. define the MDDBs either by loading the definitions from file or by using manual entry. If possible, it is better to load the definitions from file e.g a data dictionary. If new members are added e.g. as hierarchies, use meaningful descriptors and avoid ambiguous or technical terms.

6. transfer the data from the source e.g. relational database or text files, into the MDDB. Any errors in this process will either produce empty MDDBs or MDDBs with ridiculous values.

Design of the OLAP Interface.

The OLAP interface can be developed in parallel with the MDDB as long as some data is immediately available for testing the interface. A sensible first step is to create a simple table and connect it to the data source to be used for testing.

Most of the functions provided by OLAP are necessary for successful manipulation of ESD and should be provided. Based on our experience with *Abacus*, these must include a way to add and remove dimensions to tables/graphs, rolling up and drilling down and slicing. In some cases sorting and filtering may also be required.

If the tool provides support for chaining i.e. combining a number of actions, it is advisable to use it. Failing this, the developer may have to write code to control the interaction and coded instructions are more time consuming and more difficult to maintain.

Place control buttons in a fixed screen position e.g. a menu bar because the screen may contain a large table and control buttons can be lost from view while manipulating the table.

FUTURE RESEARCH

The final goal of this research is to learn more about the abilities of casual users, especially as they interact with OLAP/MDDB systems. This goal is being pursued in an observational study of a sample of casual and novice users using *Abacus*. As described above, participants are given a brief training session and are then videotaped while doing 18 representative tasks.

The videotapes will be further analysed to answer additional research questions including:

- Does a user's performance deteriorate as the number of dimensions in a table increases?
- What OLAP functions are used to solve different types of tasks

- Are there any patterns in the order in which tools are used to complete tasks?
- If there are common errors, to what can these be attributed?
- Are differences in the performance of users related to their familiarity with ESD?

CONCLUSION

This research has demonstrated that:

- Web-based, OLAP/MDDB systems may solve many problems of casual users of ESD.
- a prototype of such a system can be used to store quite complex sets of ESD, provided that the dimensions in the data sets are organised to minimise storage and upload times.
- a generic interface can be built to access multiple sets of ESD, in a variety of settings.
- this interface can perform most tasks required by casual users in a reasonable time.
- this interface can be learned in a short time and can be used for relatively complex tasks.

Initial observations of casual and novice users working with the prototype have indicated that such users find the prototype acceptable. Further analysis of these observations will extend our understanding of the task of analysing ESD and of the needs and abilities of casual users.

REFERENCES

Bannon, L.J. (1995) "From human factors to human actors: the role of psychology and human-computer interaction studies in system design", in R.M. Baecker, J. Grudin, W.A.S. Buxton & S. Greenberg (eds.), *Reading in Human-Computer Interaction: Towards the Year 2000,* Morgan Kaufmann, San Francisco.

Burgess C. and Swigger K. (1986) A graphical database interface for casual, naive users, *Information Processing and Management*, 22 (6), 511-521.

Codd E.F. (1974) Seven steps to RENDEZVOUS with the casual user, in J.W. Klimbie & K.L. Koffeman (eds.) *Data Base Management*, North-Holland.

Codd, E.F., Codd, S.B. and Salley, C.T. (1993) Providing OLAP to User Analysts: an IT Mandate, URL http://www.hyperion.com/whitepapers.cfm.

Cuff, R. N. (1980) On casual users, *International Journal of Man-Machine Studies*, 12, 163-187.

Davis, F.D. (1989) Perceived usefulness, perceived ease of use, and user acceptance of information technology, *MIS Quarterly*, September 1989, 319-340.

Eason K. (1975) Understanding the naive computer user, *The Computer Journal*, 19 (1) 2-7.

Fenichel, C.H. (1979) Online searching: measures that discriminate among users with different types of expectations, *Journal of the American Society for Information Science*, January 1981, 23-32

Fisher J. (1991) Defining the novice user, *Behaviour & Information Technology*, 10 (5), 437 - 441.

Hyland, P.N. and Hasan, H. (1997) The Use of External Statistical Data in Local Government, *Urban Policy and Research Journal*, 15 (4), 279 - 289.

Jenkins, A.M. (1982) *MIS Decision Making Performance*, Ann Arbor, UMI Research Press

Leavitt, H.J. (1965) "Applied Organizational Change in Industry: Structural, Technological and Humanistic Approaches" in March, James (ed.) *Handbook of Organizations*, Rand McNally, Chicago.

Lough D.E. and Burns A. D. (1977) An analysis of data base query languages. *Master's Thesis. Naval Postgraduate School*, Monterey, California.

Nunamaker, J.F. Chen and Purdin (1991) Systems Development in Information Systems Research, *Journal of Management Information Systems*, 7 (3) 89-106.

OLAP Council (1997) OLAP Council White Paper, URL http://www.olapcouncil.org/research/whtpapco.htm.

Ozsoyoglu, G. and Ozsoyoglu, Z.M. (1985) Statistical Database Query Languages, *IEEE Transactions on Software Engineering*, 11(10) 1071 -1081.

Pendse, N. (1999) The OLAP report: what is OLAP?, URL http://www.olapreport.com/fasmi.htm.

Preece, J., (with Rogers Y., et al) (1994) *Human Computer Interaction*, Addison Wesley, Reading, MA.

Rafanelli, M. (1989) Research Topics in Statistical and Scientific Database Management: the IV SSDBM, *Proceedings of the fourth International Conference on Statistical and Scientific Database Management*, 1-18.

Shneiderman, B. (1995) “A taxonomy and rule base for the selection of interaction styles” in R.M. Baecker, J.Grudin, W.A.S. Buxton & S. Greenberg (eds.), *Reading in Human-Computer Interaction: Towards the Year 2000*, San Francisco, Morgan Kaufmann

Vogel, D. and Nunamaker J. (1990) Group decision support system impact: multi-methodological exploration, *Information and Management*, 18, 15-28.

Weiser, M. (1993) Some computer science issues in ubiquitous computing, *Communications of the ACM*, 36 (7), 74-84.

Wynekoop, J.L. and Conger, S.A. (1991) “A Review of computer aided software engineering research methods” in Nissen, Klein and Hirscheim, (eds.) *Information Systems Research: Contemporary Approaches and Emergent Traditions*, Elsevier, Amsterdam.

COPYRIGHT