ON WRITING A 'COMPREHENSIVE', 'INTERACTIVE', 'PORTABLE', 'DATA BASE ORIENTED' AND 'RELIABLE' PROGRAM SYSTEM FOR ECONOMETRIC MODELING AND CORPORATE PLANNING (INTERIM REPORT)

Klaus PLASSER, Harald SONNBERGER,
Kurt RODLER and Wilfried PHILIPP

Forschungsbericht/
Research Memorandum No. 169

January 1982

This research was supported by the Austrian Science Foundation (Fonds zur Förderung der wissenschaftlichen Forschung) by grant No 4012. Project title: Software Development in Econometrics (Software-Entwicklung in der Ökonometrie)

Abstract:

The IAS-SYSTEM was designed and developed in order to provide a comprehensive software system for econometric modeling and corporate planning.

This paper discusses the main objectives, problems and problem solutions in connection with the production of this software. The main topics are interactive systems (man-machine dialog, analysis of dialog, error handling), portability (standardization of programming languages, standard-conforming programs, semi-portable and non-portable modules), data organization (unique interface between application programs and data base, storage, retrieval and update of items, self-reorganizing data base) and reliability (design, structured programming, documentation, test).

Kurzfassung:

Das IAS-SYSTEM wurde entworfen und implementiert, um ein umfassendes Programm-System für die ökonometrische Analyse und für die strategische Unternehmensplanung zur Verfügung zu stellen.

Dieses Paper diskutiert die wichtigsten Ziele, Probleme und Problemlösungen in Zusammenhang mit der Entwicklung dieser Software. Die wichtigsten Themenbereiche sind interaktive Systeme (Mensch-Maschine Dialog, Analyse des Dialogs, Fehlerbehandlung), Portabilität (Standardisierung von Programmiersprachen, standardgerechte Programme, halb-portable und nicht-portable Module), Datenorganisation (eindeutige Schnittstelle zwischen den Anwendungsprogrammen und der Datenbank, abspeichern, wiederfinden und verändern der Datenbankinhalte, selbstreorganisierende Datenbank) und Zuverlässigkeit (Entwurf, strukturiertes Programmieren, Dokumentation, Test).

CONTENTS

Preface

This paper has four objectives:

1) This paper is an interim report to the Austrian Science Foundation (Fonds zur Förderung der wissenschaftlichen Forschung) who has supported these investigations by grant No. 4012. It was one of the main objectives of the Austrian Science Foundation to complement the actual design and implementation of the new release of our econometric software system by scientific publications. This is an interim report showing the directions of our investigations.

2) This paper will be presented at the Second International IAS-SYSTEM Working Conference in January 1982. It should be the basis of cooperation between various institutions who are either going to implement the IAS-SYSTEM on a special computer or to add additional features to the System.

3) This paper should be a basis of discussion for groups who work in similar fields, e.g. produce similar software.

4) This paper should give users of the IAS-SYSTEM a better feeling what really happens if they employ the IAS-SYSTEM for their estimations, simulations etc.

The authors wish to express their thanks to the numerous experts from inside and outside the Institute for their assistance and recommendations. We would very much appreciate critics and comments.

Wien (Vienna), January 1982

IAS-SYSTEM Project-Team
Institute for Advanced Studies

1. INTRODUCTION

1.1. The Institute for Advanced Studies and the IAS-SYSTEM

The IAS-SYSTEM (Inter-active Simulation System) has been written with extensive use of the infra-structure of the Institute for Advanced Studies in Wien (Vienna), Austria (for more information see FÜRST 1981) and its computer center, the Computer Center for Social and Economic Research.

The objectives of the Institute for Advanced Studies are research and postgraduate education in the social sciences with special emphasis on empirical and mathematical methods. The fields covered by the departments of the Institute are (1) Management Science and Operations Research, (2) Economics, (3) Mathematical Methods and Computer Science, (4) Political Science and (5) Sociology.

Problem solving in the area of social science requires increasingly the cooperation of specialists in several fields. The Institute therefore emphasizes interdisciplinary teaching and research activities.

One of the results of this interdisciplinary research is the IAS-SYSTEM, the Interactive Simulation System, a software package (dialog and data base system) for quantitative research in economics, especially econometric modeling, and corporate planning.

Although the IAS-SYSTEM was initially only a project of the two departments (1) Economics and (2) Mathematical Methods and Computer Science, it is currently used and inspired by all five departments.

Besides the use within the Institute the IAS-SYSTEM has been implemented worldwide. In July 1981 the IAS-SYSTEM was installed 22 times in 10 countries in 4 continents (see appendix D).

1.2. Summary of the Features of the IAS-SYSTEM

The IAS-SYSTEM was designed and developed in order to provide a comprehensive software system for econometric modeling and corporate planning.

Data, equations (= algebraic expressions), models (= systems of equations) and text are stored in a data base (permanent working area).

Various data entry, update and transformation features are available. The IAS-SYSTEM offers a wide range of estimation procedures, including single equation, instrumental variable, simultaneous and system estimators. Seasonal adjustment is implemented. The results of the estimation procedures can be stored in equations or submodels (system estimators). These equations and submodels can be combined to models. The models are used for forecasting, ex post simulation, model validation and policy simulation. Two optimization procedures (linear and quadratic programming) are implemented. Various table handling programs provide report generation facilities which are accompanied by graphic output routines (for more information about the features of the IAS-SYSTEM see the summary of current documentation in appendix B).

Unlike similar systems, the IAS-SYSTEM is interactive. It employs an easy dialog language. The IAS-SYSTEM is designed for use by economists and business analysts and not for use by computer specialists.

1.3. Current Status of the Project

1.3.1. Level 2 of the IAS-SYSTEM ('IAS-2.xx')

Level 2 is the currently implemented, tested, documented and distributed version of the IAS-SYSTEM. However, this Level has one extreme drawback: it is written in a 'dialect' (= superset) of the programming language FORTRAN. This dialect provided numerous additions to the ANSI-standard of FORTRAN valid at that time (ANSI 1966). When Level 2 was designed in 1974 it was not intended to distribute the System and so Level 2 of the IAS-SYSTEM was written in this dialect, fully utilizing the efficient code generated by this compiler for our hardware.

It was very easy to implement the System on similar hardware, but it was very difficult to implement it on different hardware/different operating systems. This led to Level 3 of the IAS-SYSTEM (see below).

1.3.2. Level 3 of the IAS-SYSTEM ('IAS-3.xx')

Level 3 of the IAS-SYSTEM was designed in 1980. One of the main objectives of Level 3 is portability (see chapter 4: portable program systems).

Programming of this Level was started in December 1980. Currently January 1982), some eight commands, the data base and syntax analysis have been implemented, as well as most of the supporting functions and subroutines like conversion programs, file handling utilities, input and output modules for the different files, exception handling routines, programs for retrieving, encoding and decoding date and time, routines for setting and testing internal switches, debugging facilities etc.

These practical implementations are supplemented by theoretic research in the corresponding fields, which is funded in part by the Austrian Research Foundation (Fonds zur Förderung der wissenschaftlichen Forschung).

Main fields of research are therefore portability, programming language standardization, interactive systems, man-computer dialog, data base organization, program design, implementation techniques and management of software projects.

Most of these topics are discussed in the next chapters. These chapters begin with an overview of available techniques and literature and end with the design and implementation as used for the IAS-SYSTEM.

2. COMPREHENSIVE PROGRAM SYSTEMS

Soon after the pioneering days of the first software developers programs were collected into well documented subroutine libraries in order to be reused for further programs. The drawback of a subroutine library is that the user has to write his own main program for every single problem and that he has to combine (bind, link, collect) the main program with one or more subroutines of the program library. Typically these subroutine libraries are not used by economists, sociologists etc. but by computer experts. The economist describes his problem to a computer specialist who translates this problem into a language understandable by the computer. This prevents the direct interaction of the person who originally had the problem, e.g. an economist, and the computer.

The next step towards ease of use are comprehensive application programs, e.g. the widespread statistical packages SPSS (NIE et al. 1975) and BMDP (DIXON et al. 1979) and the econometric packages TROLL (TROLL SERIES DOO94, 1981) and TSP (HALL and HALL 198O).

The IAS-SYSTEM was designed to be a comprehensive system for the various steps of the econometric modeling process, e.g. data handling, estimation, model building, model validation, forecasting, policy simulation, report generation, plotting etc. (for details see appendix C: the IAS-SYSTEM at a glance).

A comprehensive system must have a

- unique user interface
- unique command language
- unique data base
- unique programming style
- unique documentation.

3. INTERACTIVE PROGRAM SYSTEMS

One of the inconveniences of most of the programs mentioned in chapter 2 - and of many other programs currently available - is that these programs are batch oriented rather than dialog oriented. That means that the input requested from the user has a fixed format, having punched cards in mind. These programs usually terminate in case of any wrong specification of the user's input. A reasonable man machine dialog is not possible. Typically these programs are again not used by the expert who has to solve a problem (e.g. an economist), but by a computer expert who does not know much about the original problem.

The solution to this inconvenience are interactive systems or dialog systems. There are various techniques how to organize the man-machine interaction (LOCKEMANN and MAYR 1978, MARTIN 1973, SPRUTH 1977).

Martin (MARTIN 1973, p. 87 f.) enumerates 23 techniques of conversation with the computer. However, these techniques can be combined to three classes which will be discussed in the following paragraphs:

- menu technique
- question and answer
- simple command language.

A main aspect in the judgment of these different techniques is how they can satisfy the different types of persons who are potential users of a program system.

The different types of users include

- casual users
- expert users
- parametric users (users who only enter parameters requested by the computer program)
- researchers in various fields.

3.1. Menu Technique (Menu Selection)

The description of this technique includes similar ways of dialog with the computer like form-filling, over-writing etc.

Menu technique is appropriate if there is only a very limited set of valid operator actions to a computer initiated question. In this case, these may be listed and the operator asked to select one from the "menu" of possible choices.

Menu technique can be materialized by simple WRITE/READ operations or with the help of some full screen mode.

Advantages of menu techniques:

- good for casual users, parametric users
- easy to handle

Disadvantages of menu techniques:

- boring for expert users
- high data transmission cost and time consuming if data transmission rate is slow
- full screen is not portable, but it is highly dependent on the operating system and the special hardware. It is likely to be inoperable after change of the terminals or the operating system.

3.2. Question and Answer

The question-and-answer technique is similar to menu selection but the user must know or guess the set of valid answers. Otherwise the answer will be rejected and the same question will be asked again. An intelligent computer program could change from question and answer to menu selection if the operator had transmitted an invalid response to the question asked by the computer. Another way would be to implement a HELP command which could be transmitted by the user if he does not know how to answer the question of the computer.

Note, that similar to the menu technique every action of the dialog is initiated by the computer, namely by the question of the dialog module.

Advantages of question and answer:

- relatively good for casual users
- fully portable if materialized by WRITE/READ.

Disadvantages of question and answer:

- boring for expert users
- too difficult for parametric users if the set of possible answers is large
- data transmission is lower compared to menu selection but still high.

3.3. Simple Command Language

If the command language approach is employed, the user has to learn the syntax and the semantic of the special language. Now the human user of the computer has to initiate the dialog by typing a certain command.

If this command is within the set of the allowed commands and if all other parameters are within their ranges, the computer will execute the command; otherwise it will print an error message and possibly proceed with question and answer or menu technique. Another feature to support a user in difficulties is the implementation of a powerful HELP command.

Advantages of employing a command language

- no unnecessary data transmission
- good for expert users
- low data transmission necessary from both sides
- fully portable if materialized with READ/WRITE and if the programming language has appropriate character handling facilities.

Disadvantages of employing a command language:

- syntax and semantics of the command language have to be learned
- normally a human being does not talk in a command language. Why should he/she talk to the computer in a command language?

3.4. Dialog in the IAS-SYSTEM

When designing the man-machine dialog for Level 3 of the IAS-SYSTEM the following facts were known from the usage of Level IAS-2.xx:

- Usually the users of the IAS-SYSTEM are expert users, or will become expert users soon, if they specify, estimate, manipulate and validate a model of, say, 200 equations.

- Short response time is very necessary for an interactive system. Sometimes users of the IAS-SYSTEM work with very slow data transmission rates (e.g. dial up line). Note that a line with 300 bits per second allows less than 40 characters per second. That means that more than two seconds are needed to transmit a full line of 80 characters or nearly a minute to transmit a full screen of 24 (full) lines or half a minute for a menu which covers half a screen.

- Portability is one of the main objectives of Level 3 of the IAS-SYSTEM.

For all these reasons it was decided that the IAS-SYSTEM should employ a simple (!!) command language which is supported by a question-and-answer mechanism in some cases and by a powerful HELP command.

However, the question to be answered is: "what is a 'simple' command language?":

According to the suggestions of the users of Level IAS-2.xx and according to the experience of that Level of the IAS-SYSTEM a command of Level IAS-3.xx has the following form:

```
*command,option field1,field2,...
```

A field may be divided into subfields:

```
field: = subfield1 : subfield2 : ...
```

In most cases, however, an actual command needs only one or two fields. Options and fields (subfields) are consistent in many commands (classes of commands). Expert mode suppresses additional questions by the computer. A special switch avoids page advance and/or printout of empty lines.

Examples for constant options and fields:

```
*SER,I identifier
*EQU,I identifier
*MOD,I identifier
```

These commands initiate time series, equation or model input, respectively. After these commands are transmitted the computer solicits input of the title (head line) of the item to be stored into the data base. Evidently, this question by the computer is suppressed if expert mode is 'ON'.

After input of the title the computer solicits input of the data, equation string or model components, respectively.

That means that the process is initiated by the human user, but afterwards the computer takes over the command position.

Most fields and subfields have a default value if they are not specified by the user. Unsignificant field and subfield separators (',' and ':') need not be written.

Example:

```
* DB BASE1:RDKEY:WRKEY,2000          full DB-command
* DB BASE2                           minimal DB-command
* DB BASE3,4000
```

Note, that the field and subfield separators, like other special characters, are implementer defined parameters which can easily be changed by the implementer.

3.5. Syntax Analysis in the IAS-SYSTEM

A section of the final report will be devoted to syntax analysis. This subchapter should only give a short overview.

The syntax analysis is performed in two steps:

Step 1: (command independent):

Step 1.1: decides type of line (command line, data line, scalar definition, comment line, page advance etc.)

Step 1.2: partitions command string into
- command
- option
- fields
- subfields

Step 2: Tests command dependent the types of the subfields, e.g.
- string (unchecked)
- integer
- real
- identifier
- time definition

The input is immediately prompted by the computer, especially if an error condition occurs. Error messages are full length sentences in English (German etc., see below).

If an error occurs, execution remains at the same place in the program and loops until the correct input is transmitted. Certainly there is an easy way to escape from this loop by simply typing a new command which begins with an asterisk '*' (or by typing an asterisk '*' only).

3.6. Additional Features Supporting the Dialog

All messages are stored on a message file. This message file can be translated in order to get a German, Italian, Spanish or other IAS-SYSTEM. An interactive utility program is provided to edit this message file.

There is only

1) one READ (from standard input unit)
2) one WRITE (to standard output unit)

in the IAS-SYSTEM. This programming style provides useful features:

ad 1)
Because of the fact that there is only one READ statement from the standard input unit it is very easy to implement an input log file where all input to the System is stored. This facility can be used when searching (supposed?) system errors. Additionally, a complete session or parts of a session can be repeated.

The creation of the input log file or of parts of the input log file can be suppressed by the input log switch, one of the twenty software switches of the IAS-SYSTEM.

Another advantage of this unique READ statement is the easy way to implement echo mode where all input to the System is echoed. Echo mode is especially useful for batch runs. Echo mode can again be turned on or off by a software switch.

ad 2)
Because of the fact that there is only one WRITE statement to the standard output unit it is very easy to implement an output log file where all output of the System is stored. This facility can be used to send the same output to the terminal as well as to the line printer. A whole session or parts of a session can be printed off line after inspecting the output from the terminal.

Again creation of the output log file or of parts of the output log file can be suppressed by the output log switch, another software switch.

The following software switches are currently implemented in the IAS-SYSTEM:

1	input log
2	echo mode
3	walk back in case of system errors
4	debug mode
5	output log (0,1,2)
6	error message if output is truncated
7	expert mode
8	quantity of print output (0,1,2,3,4,5,6)
9	batch mode
10	assign only minimal number of files
11	question mark mode
.	
.	
19	message file assigned
20	data base assigned

Most of these switches can only have two values (ON/OFF, 0/1), some may have more.

Currently the IAS-SYSTEM uses eight files with different access methods; some are only input files, some are only output files and some are used for both input and output:

File 11 I/O	D/A	data base: IAS-files, B*-tree
File 12 I/O	D/A	data base: numeric strings
File 13 I/O	D/A	data base: character strings
File 14 I/O	D/A	scratch-file to gain virtual memory
File 15 I/O	sequ	interface to operating system files
File 16 I	D/A	message file
File 17 O	sequ	input log
File 18 O	sequ	output log

4. PORTABLE PROGRAM SYSTEMS

4.1. Selection of the Programming Language

It is one of the main objectives of this project to write a portable program. The motto is: one software system for many hardware systems. Researchers in economics or corporate planning should be able to use the same program system and to communicate with each other, no matter which hardware or operating system they employ.

It was therefore decided that a programming language should be used, for which an American and an International standard exist (or is short before approval).

For a comprehensive description of programming language standardization see Hill and Meek (HILL and MEEK 1980).

As the American National Standards Institute (ANSI) originated most standards which were later adopted by the International Standards Organization (ISO) we usually quote ANSI as the author of the standard. Nevertheless for our final decision we claimed both ISO and ANSI standards.

In 1980 there were ANSI standards for only four general purpose programming languages, namely

ANSI X3.9-1978	FORTRAN
(ANSI X3.9-1966	FORTRAN)
ANSI X3.23-1974	COBOL
ANSI X3.53-1976	PL/I
ANSI X3.6O-1978	Minimal BASIC

There were no standards, neither national nor international for PASCAL, ALGOL, APL etc. Other languages like ANSI X3.37-1977 Programming Language APT or ANSI X11.1-1977 Programming Language MUMPS are not considered because of their special purpose.

In spite of the PL/I standard there were not many efficient PL/I compilers available in 1980. Minimal BASIC really seemed to be too minimal (file handling etc.) and too slow. COBOL does not seem to be too efficient for extensive calculations as they were expected in the IAS-SYSTEM. Therefore FORTRAN was the only language to remain in the competition. As the old standard was already obsolete and as the new language had many enhancements, the project team finally decided to use FORTRAN 77 according to ANSI X3.9-1978 and ISO 1539-1980(E) (ANSI 1978a, ISO 1980, BALFOUR and MARWICK 1979).

Advantages of FORTRAN 77:
- widespread language
- many efficient compilers
- new features like if-then-else, character handling etc.
- file and input/output handling well defined in the standard.

Disadvantages of FORTRAN 77:
- not favoured by the university community
- loop constructs incomplete
- no data structures

4.2. Internal Standards to Achieve Portability

It should be noted that ANSI X3.9-1978 is a permissive standard, that means that this standard defines a standard-conforming program (and not a standard conforming compiler) (ANSI 1978a).

A processor (compiler or interpreter together with an operating system and a hardware system) conforms to this standard if it executes standard-conforming programs in a manner that fulfills the interpretations of the standard. A standard-conforming processor may allow additional forms and relationships provided that such additions do not conflict with the standard forms and relationships.

Note that a standard-conforming program must not use any forms or relationships that are prohibited by this standard, but a standard-conforming processor may allow such forms and relationships if they do not change the proper interpretation of a standard-conforming program.

Obviously any processor has to use some extensions in order to communicate with the operating system. In Level IAS-3.xx usage of these extensions has been avoided whenever possible.

Nevertheless the code of the IAS-SYSTEM is now divided into three well-defined parts of unequal size (see HAUER 1980):

1) the portable part according to ANSI X3.9-1978
2) the semi-portable part: not portable, but portable calling sequence
3) the non-portable part: machine dependent routines.

The names of all semi-portable routines begin with letter 'Y', the names of all non-portable routines begin with letter 'Z' and all portable routines begin with the other letters in order to be able to easily separate these sets of programs.

Example of semi-portable routines:

SUBROUTINE YOPEN (...)

This subroutine opens files according to the standard but it has additional parameters in order to specify the number of records for direct access files, read key, write key etc.

SUBROUTINE YDATIM (...)

This subroutine provides date and time, both in character format and encoded in an integer variable. This program calls the non-portable subroutine ZZDATE (see below) in order to obtain date and time from the operating system.

Example for a non-portable routine:

SUBROUTINE ZZDATE (...)

Provides date and time in the special internal format of the individual computer. This program is called by the semi-portable subroutine YDATIM (see above).

Note that, according to internal conventions of the project, a non-portable program must not be called by a fully portable routine, but only by a semi-portable or by another non-portable program.

4.3. Additional Portability Problems

The character set and its representation in the computer is machine dependent, although an ANSI and ISO standardized character set exists (ANSI 1977).

For that reason all special characters (e.g. separators) are stored in a special common block and can be changed if a character is not available on a certain machine. This does not really solve the problem, however, as the motto "one software system for many different hardware systems" is violated in this case. Commands and/or models will look differently for different character sets.

Another portability problem is that the word length of a computer is machine dependent.

The standard (ANSI 1978a) defines two types of storage units:

- numeric storage units for integer, real, logical (occupying one numeric storage unit) and double precision and complex numbers (occupying two numeric storage units)
- character storage units for characters.

The standard does not specify how many bits a numeric storage unit or a character storage unit contains and it does not specify any relationship between a numeric storage unit and a character storage unit. In fact, mixing of character and numeric storage units is explicitly not allowed in many cases.

Note that a standard-conforming program must not associate character and numeric storage units in COMMON, EQUIVALENCE or the like. A standard-conforming processor may, however, allow this association according to its internal representation of numeric and character storage units, e.g. a byte machine may allow association of a numeric storage unit with four character storage units. A program using this relationship is not standard-conforming, however.

The IAS-SYSTEM does not use any of these relationships, although the current implementation of Level 3 relies on the assumption that the word length is at least 30 bits plus sign bit(s). This brings problems for the implementation on micro-computers but otherwise too much overhead would be produced. There will be a special implementation for 16-bit computers.

Quite a number of conversion programs had to be written in order to guarantee portability. Note the overhead which must arise from those conversions while short response times are a main objective of the project.

Another problem arising from the fact that word length (exactly: the length of a numeric storage unit) is not defined in the standard is floating point arithmetic. The results of some arithmetic operations are, strictly speaking, unpredictable. This will be improved in the new standard of FORTRAN 8x (see ANSI 1981), using a new PRECISION statement.

The INQUIRE and OPEN statements are not as powerful as they should be for our application. It would be very useful to know if another program uses a file or uses a file exclusively, if it employs the file only for READ-operations or if it also WRITEs onto the file etc.

A semiportable program must be written to enhance the INQUIRE statement and it is possible that a special operating system cannot answer all these questions.

The OPEN statement has to be enhanced in similar ways: open a file READ-only, assign it exclusively, use read/write keys, specify the number of records in a direct access file etc..

The implementation group heavily uses the INCLUDE feature which is not part of the standard. This facility enables both flexibility and consistency over the different modules, especially for the association and dimensions of arrays and the association of variables, e.g. separators, status variables, switches etc.

A feature which is even more powerful than this INCLUDE statement is announced for the next ANSI standard of FORTRAN, however (see ANSI 1981). Larmouth states in his article on FORTRAN 77 portability that all compilers examined so far ... provide an INCLUDE or an INSERT statement (LARMOUTH 1981).

4.4. The Future of FORTRAN

In spite of the efforts concerning the new programming language Ada (GOOS and HARTMANIS, 1981) the American Government still heavily engages in the development of FORTRAN.

The committee X3J3 of the American National Standards Institute already produced numerous standing documents concerning the new FORTRAN standard which has the working title FORTRAN 8x (ANSI 1981).

The new standard employs a Core plus modules concept, with the Core containing the most important features of FORTRAN. One of the modules is the OFM (obsolete features module), containing old and obsolete constructs like arithmetic IF, computed GOTO, alternate RETURN, ASSIGN and assigned GOTO, statement functions etc. The Core of FORTRAN 8x consists of FORTRAN 77 modified by the deletion of the features which are included in OFM.

Another module is the LEM (language extension module). It contains the new and enhanced features of FORTRAN like new loop handling, CASE statement, new precision definitions, GLOBAL statement, array handling etc.

Other modules are the SAMs (standard application modules) like graphics or real time.

The third class of modules are the BUMs (application but unstandardized modules) which may contain local supplements. Note that these abbreviations are not yet approved proposals for FORTRAN 8x and have changed considerably thru the documents (ANSI 1981).

The "minimal" FORTRAN implementation will include all of Core FORTRAN plus the "obsolete features module (OFM)", and will accept all FORTRAN 77 programs.

In any case, it must be expected that all standard FORTRAN 77 programs will continue to be accepted during this millenium (i.e., through the year 2000) by standard-conforming processors (MEISSNER 1981).

After the year 2000 another revision of the FORTRAN standard may begin to be implemented. Then those "obsolete features" that are in fact no longer widely used may disappear from most FORTRAN implementations (MEISSNER 1981).

5. DATA BASE ORIENTED PROGRAM SYSTEMS

5.1. External View of the Data Base

Normally the user of a program system wants to store his data on-line and once for ever and retrieve it by a name rather than any index or the like.

In the IAS-SYSTEM data, equations, models and text are stored into a data base (permanent working area) on mass storage. When being stored these items receive a user-defined name by which they can also be retrieved.

There is a unique interface between the application programs (IAS-SYTEM or any other application program) and the data base (IAS-SYSTEM data base or any other data base).

The logical structure of the data base is hierarchical. The data base is divided into files, the files contain elements and the elements may have different versions.

The full form of an identifier (name) of an item in the IAS-SYSTEM is therefore

```
FILE.ELEMENT (VERSION)
```

The same element name in a different file specifies a different item, a different version of the same file.element is a different item.

Examples for calculations, specifying various items:

```
*CALC AFILE.CP/BFILE.Q
*CALC AFILE.CP/BFILE.CP
*CALC AFILE.CP/AFILE.Q
*CALC AFILE.CP(1)/AFILE.CP(2)
```

For the handling of the data base no computer or data base expert is necessary, the IAS-SYSTEM is fully self-contained. Using the new DB-command all handling operations like creation, assignment, enlargement and deletion can be performed by the user himself.

A new data base is generated with the DB-command, option C.

Example for the creation of a new data base:

```
*DB,C NEWDB,2000
```

The new data base NEWDB is to be generated (cataloged). This new data base shall have space for a maximum of 2000 items (time series, equations, models, text items), as specified in the second field of the DB-command.

For security reasons READ- and WRITE-keys may be attached to the file name in subfields two and three of field one.

Examples:

```
*DB,C SECOND:RDKEY:WRKEY,5000
*DB,C THIRD::WRKEY2,2000
*DB,C FOURTH:RDKEY,3000
```

Other options of the DB-command are used to assign an old data base (previously created by *DB,C), to free such a data base, to delete a data base etc.

5.2. Internal View of the Data Base

Internally the data organization of the IAS-SYSTEM uses standard-conforming FORTRAN direct access files.

There are two main objectives for a data base:
- fast access to a single item or to a set of items
- resistancy to errors (e.g. after a stop of the operating system)

In order to achieve these objectives the data base of the IAS-SYSTEM uses a modified B*-tree data organisation (see BAYER and McCREIGHT 1972, HÄRDER 1978, KNUTH 1972, WEDEKIND 1974). A B*-tree is a generalization of binary trees and index sequential storage. The modified B*-tree within the IAS-SYSTEM has only two levels with a standard maximum of 100 keys per level, which allows a maximum of 10 000 keys (i.e. 10 000 items) within one data base. Larger data bases can be generated, but need a few changes in the implementation.

According to this data organization it is possible to retrieve a single item with only two access operations to mass storage. This number is constant and does not depend on the load factor of the data base (exactly: only one search operation is necessary if the data base contains less than 100 items).

A third access operation is needed to retrieve the contents of the item (i.e. the information which is really searched for). Numeric information and character information is stored on different files in order to avoid portability problems (see chapter 4.3).

Automatic reorganization of the data base is provided and automatic error correction (e.g. after system stop) is attempted.

A special section of the final report will be devoted to more internals of the data organization.

6. RELIABLE PROGRAM SYSTEMS

6.1. Project Organization

The project group is organized as a chief programmer team (see BAKER 1972, MILLS and BAKER 1973). The members of the group are varying according to the phase of the life cycle.

The definition was specified in close cooperation with the department of economics. Collaboration decreased in the design phase. The implementation phase does not need close cooperation either. During the test phase cooperation increases again.

As the project group was rather small (4 - 7 members) it was easy to delegate whole complexes like data base organization, syntax analysis etc. to subgroups (management by objectives).

Besides numerous informal consultations the team gathered for an official meeting about once every ten days.

Major decisions of these meetings were collected in protocols. Each member of the project team received a copy of every protocol. One set of protocols was collected to become the master set of protocols.

More about the project organization will be written in the final report.

6.2. Design, Programming Style and Programming Conventions

According to the warnings in Brooks' book (BROOKS 1975) all available suggestions for software engineering were considered (see e.g. ENDRES 1978, GOOS and HARTMANIS 1975, KIMM, KOCH, SIMONSMEIER and TONTSCH 1979).

The whole program is divided into numerous independent subroutines with a well defined interface. Subroutines should not have more than 150 lines of code (excluding comment lines). GOTO is avoided whenever possible (see DIJKSTRA 1968) in spite of the restrictions of FORTRAN 77. IF( )THEN-ELSE IF( )THEN-ELSE-END IF constructs are always used to increase the structure of the program.

The contents of the individual blocks and the blocks of DO-loops are indented to improve readability.

Capital letters are used for FORTRAN 77 keywords, lower case letters else.

The internal program documentation must at least contain

- function of the program
- parameters (input, output, transput)
- access to global variables (COMMON)
- access to files
- documentation of the function of individual sections

Critical parts of the code are controlled by code-inspection (see FAGAN 1976). A debug switch enables the implementer to debug the System at run time. This debugging is supported by the input and output log file facilities. Test jobs are available.

The following FORTRAN 77 statements may only be used after consulting the project group at an official meeting:

- COMMON
- external READ/WRITE
- ASSIGN
- STOP
- ENTRY
- EXTERNAL
- INTRINSIC
- BLOCKDATA
- statement functions

The following FORTRAN 77 constructs should only be used with great care:

- EQUIVALENCE
- GOTO
- arithmetic IF
- DATA
- transput variables as dummy arguments
- constants as actual arguments
- mixing character and numeric variables when writing on mass storage

Tricky programming and bit-fiddling has to be avoided whenever possible!

SUBROUTINES may have an alternate return specifier which is always used as error return. The next parameter (i.e. the first, if no alternate return specifier is used or the second, else) is an error code. It is positive in case of a severe error, negative in case of a minor error or zero in case of no error.

The next parameters in the dummy argument list are input parameters, followed by output parameters, followed by transput parameters (parameters which may be both input and output).

Portable, semi-portable and non-portable routines are carefully separated and collected in well-defined sets of programs. The members of these sets can be distinguished by the first letters of their names. Analogously, subroutines and function subprograms are separated and within the set of function subprograms there are three subsets, namely (1) integer and logical functions, (2) real, double precision and complex functions and (3) character functions.

The PARAMETER statement is used extensively in order to combine flexibility and consistency. During program development the non-standard INCLUDE statement is employed for the same reason. For distribution the INCLUDE statement is replaced by the actual contents of the procedure to be included.

APPENDIX

References:

American National Standards Institute, Inc. (ANSI, ed.): ANSI X3.9-1966 American National Standard Programming Language FORTRAN. New York 1966

American National Standards Institute, Inc. (ANSI, ed.): ANSI X3.23-1974 American National Standard Programming Language COBOL. New York 1974

American National Standards Institute, Inc. (ANSI, ed.): ANSI X3.53-1976 American National Standard Programming Language PL/I, New York 1976

American National Standards Institute, Inc. (ANSI, ed.): ANSI X3.4-1977 American National Standard Code for Information Interchange. New York 1977

American National Standards Institute, Inc. (ANSI, ed.): ANSI X3.9-1978 American National Standard Programming Language FORTRAN. New York 1978a

American National Standards Institute, Inc. (ANSI, ed.): ANSI X3.60-1978 American National Standard Programming Language Minimal BASIC. New York 1978b

American National Standards Institute, Inc. (ANSI, ed.): Proposals Approved for FORTRAN 8X. X3J3/S6.80. New York 1981

BAKER, F.T.: Chief Programmer Team Management of Production Programming. IBM System J. 11(1), 1972

BALFOUR, A. and D.H. MARWICK: Programming in Standard FORTRAN 77. London 1979

BAYER, R. and E. McCREIGHT: Organization and Maintainance of Large Ordered Indices. Acta Informatica 1(3), 1972

BROOKS, F.P.: The Mythical Man Month. Reading 1975

DIJKSTRA, E.W.: Go to Statement Considered Harmful. Comm. ACM 11(3), 1968

DIXON, W.J. and M.B. BROWN (ed.): BMDP-79. Biomedical Computer Programs. P-Series. Berkeley 1979

ENDRES, A.: Methoden der Programm- und Systemkonstruktion: Ein Statusbericht (German). In: SCHINDLER, S. and W.K. GILOI: GI-8. Jahrestagung. Berlin-Heidelberg 1978

FAGAN, M.E.: Design and Code Inspections to Reduce Errors in Program Development. IBM Syst. J. 15(3), 1976

FUERST, E.: Institute for Advanced Studies Vienna. Wien (Vienna) 1981

GOOS, G. and J. HARTMANIS (ed.): Software Engineering. An Advanced Course. Berlin-Heidelberg 1975

GOOS, G. and J. HARTMANIS (ed.): The Programming Language Ada. Reference Manual. Proposed Standard Documentation. Berlin-Heidelberg 1981

HALL, B.H. and R.E. HALL: Time Series Processor. Version 3.5. User's Manual. Stanford 1980

HAUER, K.-H.: Portable Methodenmonitoren (German). Berlin 1980

HÄRDER, T.: Implementierung von Datenbanksystemen. München-Wien 1978

HILL, I.D. and B.L. MEEK (ed.): Programming Language Standardisation. Chichester 198O

International Standards Organization (ISO ed.): ISO 1539-1980(E). Programming Language FORTRAN. Sine loco 1980

KNUTH, D.E.: The Art of Computer Programming, Vol. 3: Sorting and Searching. Reading 1972

LARMOUTH, J.: Fortran 77 Portability. Software-Practice and Experience 11(10),1981.

LOCKEMANN, P.C. and H.C. MAYR: Rechnergestützte Informationssysteme (German). Berlin 1978

MARTIN, J.: Design of Man-Computer Dialogues. Englewood Cliffs 1973

MEISSNER, L.P.: Core and Modules (Editorial Comments). In: MEISSNER, L.P. (ed.): Fortran Newsletter 7(4), 1981

MILLS, H.D. and F.T. BAKER: Chief Programmer Teams. Datamation 19(12), 1973

NIE, N.H., C.H. HULL, J.G. JENKINS, K. STEINBRENNER and D.H. BENT: SPSS. Statistical Package for the Social Sciences. Second Edition. New York 1975

SPRUTH, W.G.: Interaktive Systeme (German). Stuttgart 1977

TROLL SERIES: DOO94. TROLL Bibliography. Cambridge (M.I.T., Information Processing Services) 1981

WEDEKIND, H.: On the Selection of Access Paths in a Database System. In: KLIMBIE, J.W. and K.L. KOFFEMAN (ed.): Data Base Management. Amsterdam 1974.

Summary of Current Documentation (January 1982, Level IAS-2.16 and IAS-3.1)

K. Plasser H. Sonnberger	IAS-SYSTEM, Level IAS-2.16, Installation Guide and Implementer Manual. Institutsarbeit Nr.148, Institute for Advanced Studies, Vienna(Austria) May 1981
K. Plasser	IAS-SYSTEM, Level IAS-2.16, BRIEF DESCRIPTION. Institutsarbeit Nr.143, Institute for Advanced Studies, Vienna (Austria), February 1981
K. Neusser H. Sonnberger	IAS-SYSTEM, Level IAS-2.16, USER REFERENCE MANUAL - PART TWO. Institutsarbeit Nr.140, Institute for Advanced Studies, Vienna (Austria), December 1980
K. Rodler	IAS-SYSTEM, Level IAS-2.15, UPDATE TO USER REFERENCE MANUAL - PART ONE (Level IAS-2.14). Institutsarbeit Nr.138, Institute for Advanced Studies, Vienna (Austria), November 1980
K. Plasser	IAS-SYSTEM, Level IAS-2.14, USER REFERENCE MANUAL - PART ONE. Institutsarbeit Nr.129, Institute for Advanced Studies, Vienna (Austria), July 1980
K. Plasser	IAS-SYSTEM, Level IAS-2.14, PROSPECTUS. Institutsarbeit Nr.123, Institute for Advanced Studies, Vienna (Austria), February 1980
R. Matuschek K. Plasser I. Prucha	Interface between the IAS and the LINK-SYSTEM. Internal Discussion Paper No. 3/1978, Institute for Advanced Studies, Vienna (Austria), March 1978
K. Plasser	Vermarktung des IAS-SYSTEMS (bisherige Erfahrungen und (Miss-)Erfolge) (German). Institutsarbeit Nr. 151, Institute for Advanced Studies, Vienna (Austria), June 1981.
K. Plasser	Anwendung des IAS-SYSTEMS fuer den oekonometrischen Modellbau und fuer die modell- und computer-unterstuetzte Unternehmensplanung. Instututsarbeit Nr. 155, Institute for Advanced Studies, Vienna (Austria), November 1981.

K. Plasser H. Sonnberger K. Rodler W. Philipp	On Writing a ´Comprehensive´, ´Interactive´, ´Portable´, ´Data Base Oriented´ and ´Reliable´ Program for Econometric Modeling and Corporate Planning (Interim Report). Forschungsbericht/Research Memorandum No. 169, Institute for Advanced Studies, Vienna (Austria), January 1982.

THE IAS-SYSTEM AT A GLANCE

- Dialog-oriented System
  - Batch mode
  - Interactive mode
    - Easy command language
    - Immediate error detection
    - System answers and error messages in English
- Data-management
  - Integrated data-base
    - Time-series (=vectors)
    - Equations (=algebraic strings, see below)
    - Models (=systems of equation, see below)
  - Data-manipulation
    - Automatic aggregation
    - Vector-algebra
      - Addition, subtraction
      - Multiplication, division (component by component)
      - Multiplication, division (by scalar)
      - Elementary functions
        - SIN, COS, EXP, LOG
      - Scalar product
    - Data-transformation
      - Differences
        - Absolute differences
        - Relative differences
      - Exponentiation, logarithm
      - Lags
    - Data-generation
      - Random number gererator
      - Recursive process
      - Input from terminal, card deck
      - Results from different operations
        - Estimation
          - Residuals
          - Calculated time-series
        - Seasonal adjustment
          - Seasonal factors
          - Seasonally adjusted series
        - Calculated series
      - Interfaces to other data bases
    - Data-update
      - From terminal, card deck
      - Calculated from other series
      - In differences
        - Absolute differences
        - Relative differences
        - Differences over the current period
        - Differences over previous periods
      - As result of an equation (=algebraic string)
      - As result of a model (=system of equations)
      - Interfaces to other data-bases

```
    Output of data
       Lists
       Reports
          Without transformation
          With differences (see above)
          Calculated time-series (see vector-algebra)
       Tables (see reports)
       Scatter diagrams
          One time-series
             With or without differences
          Up to twelve time-series

Estimation
   Ordinary least sqaures
      Homogenous and inhomogenous equations
      Polynomial lag
      Autocorrelative transformation
         Hildreth-Lu
         Cochrane-Orcutt
         External specification of RHO
      Transformation of input data
         Differences
            Absolute differences
            Relative differences
         Exponentiation, logarithm
         Calculated time-series (see vector algebra)
   Instrumental variable estimation
   Simultaneous estimators
      Two stage least squares
      Limited information maximum likelihood
      K-class
   System estimators
      Two stage least squares
      Three stage least squares
      Limited information instrumental variable efficient
      Full information instrumental variable efficient

Equations (=algebraic strings)
   Different types of equations
      Behaviorial (estimation see above)
      Definitorial (=identities)
      Inequalities
      Objective functions for LP/QP
      Linear and non-linear equations
   Manipulation of equations
      Input from terminal, card deck
      Print-out
      Check of equations
```

- Models (=systems of equations)
  - Manipulation of models
    - Input from terminal, card deck
    - Update of models
    - Short print-out
    - Long print-out, including all equation strings and parameters
  - Estimation with system estimators (see above)
  - Solution of models (=equation systems)
    - Forecast, policy simulation
    - Ex post simulation
  - Call of submodels
  - Block structure of models

- Linear and quadratic programming models (=systems of equations and inequalities and an objective function)
  - Manipulation of LP/QP-models (see models)
  - Solution of LP/QP-models
    - Maximization, minimization
    - Print-out of results, slack variables, reduced costs and value of objective function
    - Update to data base

- Seasonal adjustment
  - Additive, multiplicative seasonal factors
  - Monthly data, quarterly data
  - Storage of seasonally adjusted series and seasonal factors

- Documentation (in English, computer readable)
  - Prospectus
  - Brief Description
  - User Reference Manual
    - Part One
    - Part Two: Estimation
  - Implementer Manual (installation guide)
  - Special manuals
    - Interface between the IAS and the LINK-System

- Programmer orientation (Level IAS-3.xx)
  - FORTRAN 77 (ANSI X3.9-1978, ISO 1539-1980)
    - (Level IAS-2.xx in a FORTRAN V dialect)
  - Modular
  - Highly structured
  - Direct access files
    - Data base
    - Message file
    - Virtual (main) memory
  - Sequential Files
    - Input log
    - Output log
  - Internal tables
    - File control table (FCT)
    - Time control table (TCT)
    - Status table (STATUS)

Worldwide Installations of the IAS-SYSTEM

In July 1981 the IAS-SYSTEM was installed 22 times in 10 countries in 4 continents:

AUSTRIA (4)

Bundesministerium fuer Finanzen (Federal Department of Finance), Wien

International Institute for Applied Systems Analysis (IIASA), Laxenburg

Oesterreichische Nationalbank (National Bank of Austria), Wien

SPERRY UNIVAC Austria and Comecon, Wien

FINLAND (1)

Suomen Pankki (Finlands Bank), Helsinki

GERMANY (6)

These German installations are installations of the IAS-SYSTEM Bonn. This version is based on Level 2.3 of the IAS-SYSTEM Wien and has been developed independently since 1975 at the Institute fuer Gesellschafts- und Wirtschaftswissenschaften of the University of Bonn.

Deutsches Institut fuer Wirtschaftsforschung (German Institute for Economic Research), Berlin

Fernuniversitaet Hagen (Open University of Hagen), Hagen

Gesellschaft fuer Mathematik und Datenverarbeitung (GMD) (Society for Mathematics and Data Processing), Bonn

Universitaet Berlin (University of Berlin), Berlin

Universitaet Bonn (University of Bonn), Bonn

Universitaetseminar der Wirtschaft (University Seminar of the Domestic Economy), Bonn

ITALY (3)

INDATA s.r.l. (INDATA ltd.), Roma

Istituto Nazionale per il Commercio Estero (National Institute for Foreign Trade), Roma

Universita di Roma (University of Roma), Roma

MALAYSIA (1)

Petroliam Nasional Berhad (PETRONAS) (National Petroleum Company), Kuala Lumpur

PORTUGAL (1)

Banco Espirito Santo e Comercial de Lisboa, Lisboa

SOUTH AFRICA (2)

Buro vir Ekonomiese Ondersoek (BEO) (Bureau for Economic Research (BER)), Stellenbosch

Universiteit van Wes-Kaapland (University of the Western Cape), Bellville

SPAIN (1)

Compania Espanola de Petroleos (CEPSA) (Spanish Petroleum Company), Madrid

SWITZERLAND (1)

Universite de Geneve (University of Geneva), Geneve

UNITED STATES OF AMERICA (3)

Bureau of the Census, Washington D.C.

University of Alabama, Tuscaloosa Al.

University of Maryland, College Park, MD