APPROVED FOR PUBLICATION

AERE - R 9642

17 MARS 1980

United Kingdom Atomic Energy Authority

HARWELL

Status and integrated information systems

F.N. Teskey
Computer Science and Systems Division
AERE Harwell, Oxfordshire
January 1980

CERN LIBRARIES, GENEVA

CM-P00068700

THIS DOCUMENT IS INTENDED FOR PUBLICATION IN THE OPEN LITERATURE. Until it is published, it may not be circulated, or referred to outside the organization to which copies have been sent.

C13

APPROVED FOR PUBLICATION

Enquiries about copyright and reproduction should be addressed to the Scientific Administration Office, AERE, Harwell, Oxfordshire, England OX11 0RA.

AERE - R 9642

STATUS

and

INTEGRATED INFORMATION SYSTEMS*

F. N. Teskey

ABSTRACT

This paper shows how a free text, Boolean-type retrieval system can be used to provide a wide range of information processes. The paper describes the STATUS retrieval system, concentrating on the recent developments and improvements to the system. In particular, the use of pre- and post-processors is discussed. These give much greater flexibility to the user interface and allow STATUS to be integrated into existing information systems. This is illustrated with examples from a library loans system. The paper concludes by looking at STATUS both in the context of existing information systems and also as a fore-runner to the next generation of information systems.

*This paper was first presented to the 7th Cranfield International Conference on Mechanised Information Storage and Retrieval Systems, held at the Cranfield Institute of Technology 17th - 20th July, 1979.

Computer Science and Systems Division,
AERE, Harwell.

January 1980

HL80/25 (C.13)

CONTENTS

APPENDIX

1. Introduction

In this paper we describe the STATUS program and show how it can form part of an integrated information system. STATUS is an information retrieval package that has been developed by the U.K. Atomic Energy Authority at Harwell and is now used in a wide range of applications both here and overseas. The system is described in detail in the user manual (1); in this paper we will look at STATUS from a more general point of view.

An information system has two basic components - a user and a data base. The user wants to obtain some more or less well-defined piece of information, such as the times of the trains from London to Bedford, or a list of the most important papers presented at the last Cranfield Conference. The data base is a collection of more or less well-structured pieces of information. We require, firstly that the data base contains the information the user wants, and secondly some means of retrieving that information and presenting it to the user. The latter can be considered under three headings:

(i) command language

(ii) data input and output

(iii) storage and retrieval

The last of these topics seems to have received most attention in the published literature, but we believe that in an integrated system all three functions are equally important. We will describe the basic STATUS system under each of these headings and then proceed to look at some of the more advanced features in greater detail.

2. Basic STATUS system

In this section we will outline the fundamental design of STATUS under the headings given in the previous section. It will prove convenient to take these heading in the reverse order:

(i) Storage and retrieval

In STATUS we have adopted the 'free text' principle that any text in machine readable form can be stored in the data base. The subject of the texts can be as diverse as those of Acts of Parliament or properties of hazardous chemicals, but they are all stored in the same format on the STATUS TEXT file (the main feature of this format is that the text is divided into ARTICLES). The articles in the text file are retrieved using standard inverted file techniques. STATUS automatically creates an inverted file, called the CONCORDANCE file, which contains an index to all the words in the text. Common words such as 'a', 'is', 'the' etc. need not be indexed. The concordance can then be used to retrieve articles contains a certain word or logical combination of words. This set of articles is known as the RETRIEVED LIST.

(ii) Data output

Once the concordance has been used to produce a retrieved list, the original text of each article can be output from the text file. The basic options are to output either a short title or the complete text of one or more of these articles. We will see later that the user has a number of additional options.

(iii) Command language

In STATUS we have tried to present the user with a command language that is precise and simple to use. We have adopted the basic syntax

<command> <parameters>

for the majority of commands. For example the command to list the first four titles of the retrieved list is

TITLES 1-4 (or T 1-4).

However, where this syntax could be confusing, a more natural alternative has been adopted. Thus the syntax for search questions is

Q <question body> ?

where the question body is built up from the basic logical operators

a + b (AND)

a - b (NOT)

a , b (OR).

Thus the command to retrieve articles dealing with computers and the law is quite simply

Q COMPUTERS + LAW ?

3. Advanced features of STATUS

In the previous section we have described the basic STATUS system. With this system a non-technical user can often retrieve a considerable amount of useful information. There are a number of additional features in the system to allow better use of the data in the system. These features are described below:

(i) Retrieval

The classic problem in retrieval is to increase precision and recall. There are a number of devices in STATUS to do this. The concordance file contains the exact location of each word in the text and so precision can be increased by using the COLLOCATION operator to search for words in a given position relative to each other. The most common application of this is to search for adjacent words or phrases. Precision can also be increased by the use of NAMED SECTIONS; these are user defined divisions of articles and allow a search to be restricted to certain fields. It is, for example, possible to restrict a search to a title or an abstract section. The recall of a search can be improved by the use of truncation and synonyms. STATUS allows the user to search for words beginning with a given set of characters or stem. Thus the command:

```
Q SAFE*?
```

will retrieve not only all those articles containing the word SAFE but also SAFETY, SAFELY, SAFENESS etc. The SYNONYM facility allows the data base manager to define groups of words as synonymous; a search for one of these words can then retrieve articles containing any of the synonymous words.

STATUS can retrieve articles not only using the lexical methods described above, but also using numerical range searching. Numerical data can be stored in KEY FIELDS; these consist of a name and a value e.g. # HEIGHT 1.27. The data base can then be searched for articles containing a given key field name with an associated value in a specified range e.g.

```
Q  #  HEIGHT > 1.25 ?
```

(ii) Data Output

Precision can be just as important in data output as in retrieval. Once a user has retrieved a set of relevant articles he may want to look at only a small part of each article, rather than the whole article. The named sections, introduced to increase precision in retrieval, can also be used to increase precision of data output. The user can specify exactly those sections he wants to see. There is also a CONTEXT command which outputs just those lines of text containing the words asked for in the search question. The output of numerical data can be improved by using key fields to sort the retrieved articles in

order of the key field value. Since key field values may also be character strings this allows the production of alphabetical lists.

As well as specifying exactly what data he is interested in, the user can also specify where the data should be output. STATUS provides a comprehensive facility to allow data to be routed to and from the terminal, fast printer and temporary or permanent files. We shall see later that the ability to transfer data to and from permanent files makes it very easy to integrate STATUS with other information systems.

(iii) Command language

We have seen how the retrieval and output functions have been made as flexible as possible; the same flexibility is achieved in the command language by means of a macro processor. At its simplest, the macro processor allows the user to define his own abbreviations for frequently used commands or parts of commands. This can be extended to define short programs or sequences of STATUS commands. These macros can prompt the user for various parameters to be inserted into the commands and can take appropriate action in the event of any error conditions. We shall see in the next section how these macros can be used to tailor the command language to a particular user requirement.

4. Examples

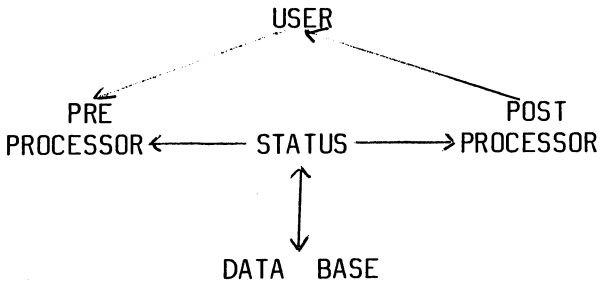
As we indicated in the introduction, STATUS is used for a wide range of applications and no single example could cover all the possibilities of the system. Some of the existing applications are listed in Appendix 1. We shall, however, attempt to illustrate some of the features of the system with examples from one of the more recent applications - the Harwell Automated Loans (HAL) system for the main library at Harwell. This application highlights three important points:

(i) STATUS can be used in unexpected areas - one would expect library applications to be in the field of cataloguing and document retrieval rather than in loans control.

(ii) STATUS can be integrated into existing systems - the system has been designed to minimise the changes needed to existing procedures.

(iii) STATUS provides power and flexibility for future enhancements.

The basic flow of information and control in the HAL system is as shown below



The data base contains the catalogue information, list of copies on loan, waiting list and any notes for each document in the system. The user can enter transactions to retrieve or update this information; examples of such transactions would be to list all the documents on loan to a given borrower, to add a borrower to a waiting list, to list overdue loans etc. The pre-processor translates these transactions into the STATUS command language, STATUS then performs the required retrieval and updating of the data base and routes the relevant information to the post-processor. The post-processor then presents the information to the user in a suitable format. Each of these processes is independent and the user need never be aware of the internal mechanism STATUS system. In the present implementation a micro-computer in the library is used to record the transactions on diskettes, the transactions are then processed overnight on the main-frame computer and a printed report sent to the library next morning. This is illustrated in the following examples:

(i) Borrower search

To request a list of all the documents on loan to a particular borrower the user would enter the following transaction (system prompts in upper case, user input is lower case).

```
TRANS      :  bs

BORROWER   :  teskey f n

DIV        :  css

REASON     :  demonstration
```

This is translated into a STATUS command to search the data base for all documents with the given borrower in the loan section. The catalogue information for each of these documents is then passed to the post-processor; this produces a report of the form:

```
Borrower  Search

To  Teskey F.N.

      CSS

Reason for search:

 demonstration

  Author/Title                    Accn. No.

Brown P J/Macro Processors        31724X

Toffer C /. . . . . .
```

(ii) Reservation

The transaction to add a borrower to the waiting-list for a document is as follows:

```
TRANS      :  r

ACCN NO    :  31724X

BORROWER   :  teskey f n

DIV        :  css
```

STATUS searches the data base for the article with the given accession number and edits the article to include the borrower in the waiting-list. This updated information is then included in the daily list of documents on loan produced by the post-processor.

5. Conclusions

Information retrieval systems tend to be classified either as document retrieval or data retrieval. Sparck Jones and Kay (6 p.175) distinguishes these two types of system as follows:

"The document retrieval system responds to a user's request for information, not by supplying the desired information itself, but by providing documents, or the descriptions of documents, in which there is some hope of finding the information. A fact retrieval system, on the other hand, aims to answer questions directly . . . it must be capable of finding facts . . . on the basis of which to infer a correct answer to the question"

This provides a useful classification of the functions of a retrieval system but it does not draw any distinction between the possible implementations of such systems. In this paper we have attempted to show how these functions have been integrated into STATUS. This integration is possible because STATUS is a free text system; if the text in the data base consists of standard bibliographic records then STATUS can function as a document retrieval system, while if the data base contains complete data records then we can retrieve that data.

There are numerous other information systems but the comparison of such systems is exceedingly difficult. So, rather than attempt to compare STATUS with all other systems, we shall consider only those features which tend to characterise and distinguish the system. Perhaps the two most important are:

(i) generality - the system can create and search data bases of any type of textual material. Thus many differing information requirements can be met by a single system, for example the library at the Transport and Road Research Laboratory uses a single STATUS data base for ordering, cataloguing and loans control. This can reduce unnecessary duplication of information in different data bases and can also reduce the effort in understanding and running several different systems.

(ii) ease of use - there is a small set of basic commands and it is possible to set up other commands to simplify the performance of more complex operations. For example the Home Office set up the Hazardous Chemicals Data base to be used by firemen with no previous computing experience, and the successful operation of the system is no doubt partly due to the simple subset of commands used. As individuals become more confident then they go on to use the more advanced features of the system.

Almost as important are:

(iii) Flexible output - the user can specify precisely what data he wants output and where the data is to be routed.

(iv) portability - the programs are written in FORTRAN and so can easily be implemented on a wide range of machines.

(v) on-line incremental update - the data base can be updated on-line and the update affects only those parts of the data base that have been changed.

We do not claim that any of these features are unique. There are, for example, numerous free text systems such as STAIRS (2) and POLYDOC (3); the latter, like STATUS, is a portable FORTRAN program. There are also systems where the user interface is so simple as to be almost transparent; these range from natural language systems such GUS (4) to special purpose systems such as the New University of Ulster's library automation (5). Usually, however, one has to choose between a general purpose system with a complex user interface or a special purpose system with a simple user interface.

In the STATUS system we have tried to combine these features to give an integrated approach to information systems. It seems certain that in the next few years there will be a great increase in the amount of information available in machine readable form, and that more and more laymen will want access to this information. To meet this demand we will need to develop a new generation of integrated information systems. They will have to cope with a wide range of information and yet be simple to operate. The STATUS package, which we have described in this paper, may well be a fore-runner to such a system.

Acknowledgements

I would like to thank Ian Croall and the other members of the STATUS group for their helpful comments and criticisms of this paper. I would also like to thank Chris Wilson and his staff in the Harwell Main Library for all their help during the development of the HAL system.

The conclusions in Section 5 are an attempt to summarise the various discussions following the original presentation of the paper. I am particularly indebted to Brian Perry and Karen Sparck Jones, for their very useful comments.

References

(1) STATUS Information retrieval system User Manual. Edited by I.F. Croall. Computer Science and Systems Division, AERE, Harwell, January 1979.

(2) Storage and Information Retrieval System (STAIRS) GH12 - 5114 - 3 IBM World Trade Corporation.

(3) Brisner, O. POLYDOC - the development of an information system for batch and on-line operation, 7th Cranfield International Conference on Mechanised Information Storage and Retrieval, 1979.

(4) Bobrow, D.G. et al. GUS, a frame-driven dialog system, Artificial Intelligence, Vol. 8, pp. 155-173.

(5) Wintour, B.J.C. and McDowell B. Automation at the New University of Ulster, Program vol. 10, No. 2, pp 60-74, April 1976.

(6) Sparck Jones, K. and Kay, M. Linguistics and Information Science, Academic Press, 1973.

APPENDIX I

STATUS : A Selection of users and applications

Name	Applications	Status*	Approx. Records	Approx. data	Computer
Attorney General's Dept. (Australia)	Legal Information System	o	40,000	80Mb	Burroughs B6700
	Departmental opinions	o	1,000	½Mb	
BNF Metals Technology Centre	BNF Abstracts & Retrieval	o	20,000	10Mb	Prime 300 (224 kb)
	Company information	o	1,000	½Mb	
BOC Ltd.	Document management	o	2,500		IBM/Amdahl (8 Mb)
	Technical Abstracts (properties of gases)	o	500	1/10Mb	
Building Research Establishment (DoE)	Library database (Books & periodicals)	o	40,000	16Mb	Prime 300 (256 kb)
	Research project data for management	o	500		
	Reports of building faults	d	6,000	12Mb	
	Inventory of instruments	p	2,000		
Commission of the European Community	CELEX (Community Law)	d	30,000	120Mb	ICL 2980 (7 Mb)
	Internals Documentation ECØ1	d	110,000	300Mb	
	Actualities	d	20,000	12Mb	
	Proposals to Council of Ministers	d	4,500	12Mb	
Department of Energy	North Sea Oil Production System	o	400	10Mb	IBM 3033 (8Mb)
Department of Health (Australia)	National drug information service	o	100	2Mb	IBM 370

* Status : o = operational; d = active development; p = planned database

European Law Centre (EUROLEX)	UK & EEC legislation, cases etc.	d	100,000+	200Mb	IBM/Amdahl (8Mb)
Home Office	Hazardous Chemicals Database	o	10,000	15Mb	IBM/Amdahl
Howson-Algraphy Group, Vickers Ltd.	Technical Abstracts	o	5,000	3Mb	ICL 2904 (96kw)
	Index of internal reports	o	700		
	Product formulae (Health & Safety)	d	150		
KLUWER GroepRechts-wetenschappen BV	Court decisions	o		80Mb	2xHB6620 (2x256kw)
	Legal literature	o		10Mb	
	Legislation	d		20Mb	
Metal Box Ltd.	Technical Reports	o/d	2,000	6Mb	ICL 1904 (32 kw)
	Library Catalogue entries	o/d	3,000	5Mb	
	Library Loan records	d			
Ricardo Consulting Engineers Ltd.	Technical Abstracts	d			ICL 2904 (96 kw)
	Engine data base	d			
Rutherford Laboratory (SRC)	Internals manuals	o	300		Prime 400 (1 Mb)
	Personal literature database	o	100		
	Engineering drawing specs.	d	11,000		
	Equipment Holdings	d	330		
Safety in Mines Research Establishment (H and SE)	Library Catalogue	o	10,000	10Mb	XEROX Sigma 6 (128k)
	Chemical Substances Database	o	12,000	12Mb	
	Accident Studies	d	1,200		
	Research data	d	500	50Mb	
	Toxic chemicals data	p	50,000		
	Factory Inspectors Report	p			

* Status : o = operational; d = active development; p = planned database

Transport & Road Research Lan (DoE)	Library ordering, catalogue loan	o	12,000	10Mb	Prime 400
	Integration with Int. database	d	60,000	150Mb	(512kb)
	Ongoing research database	o	10,000	20Mb	
	Computer programs database	o	200		
UKAEA Culham Laboratory	Database of results of Fusion R & D	p		10Mb	2xICL2976
	Library database on plasma physics	p		50Mb	(8Mb)
UKAEA Harwell Laboratory	Library Loan Control System	o	3,000	1,5Mb	IBM 3033
	Customer Information System	o	30,000	10Mb	(8Mb)

*Status : o = operational; d = active development; p = planned database