# Electronic File Cabinet -

# Online Information Systems for unstructured Data

by Franz-Josef Boll and Joachim Geffken

## Introduction

Most of today's computers store large amounts of text. These are typically unstructured data resulting out of day to day computer usage in word processing, documentation, office automation etc. Whereas data in structured files can easily be accessed by Query type software, documents, like unstructured data in general, require special retrieval techniques. This paper will present the "biography" of a document retrieval system - the package IDT - EF (Electronic File Cabinet). And it will discuss some of the general principles of document storage and retrieval, like searching, sorting, user interfaces and aspects of implementation.

IDT-EF has been developed by Franz-Josef Boll at Herbert Seitz KG of Germany in close cooperation with Radio Television Luxembourg (RTL). Although it is a general purpose software, a short overview over RTL will help to understand the scope of the software.

## Radio Luxembourg - A Large Media Company

Radio Luxemburg (Radio Tele Luxembourg) is a large private owned comercial media company in Europe. They broadcast radio and television programs on many channels for different European Languages. They are the largest private owned company of that type in Europe. RTL is also engaged in cable tv and most recently in satelite tv.

Large amounts of data, mostly unstructured, have to be handled in the administration of texts, photos, audio and video tapes and films.

## Creating a Retrieval System

One of the major challenges was the film archive: a large number of short films transmitted every day as part of the newcasts. Computer usage was necessary to handle the archive efficiently.

We met that needs by creating a document retrieval system, where the documents contain all the keywords required, secondary information and references to the audio and video information stored in different archives.

When we started working on this, we already had the experience of creating a large word processing system. From that, we had a large number of subroutines that we could use.

Very helpful to us was RTL's experience in this field. They had done extensive investigation on retrieval systems. They provided us with many ideas about the retrieval system. Thus it was possible to set up the system in a few months instead of a couple of years. **IDT-AS** has in the meantime proven that it meets the requirements of RTL as well as the requirement on a general purpose retrieval system.

**Descriptors and Selectors or How to Retrieve a Document**

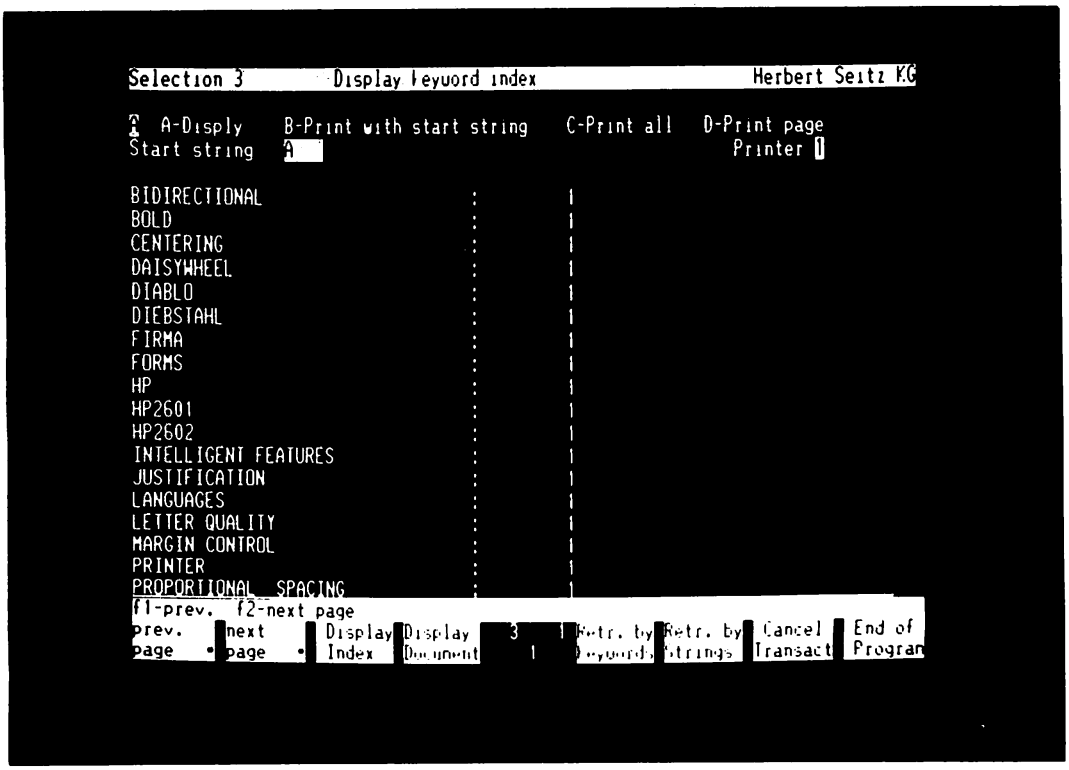
The more documents you want to store, the more useful is a document retrieval system. Large systems are only manageable with computer support. As document data bases gain substantial sizes, sophisticated retrieval algorithms are required. Even in extremely large text data bases retrieval should be fast - and accurate. That means with todays computers (RTL uses hp 3000-44's) you have to be able to retrieve a small set of documents out of 10,000 or even more. All retrieval should be done online. Response time must be in the range of a few seconds for standard and a few minutes for infrequently used transactions. There are basically three categories of information that identify a document:

- Keywords
- the sequence of characters forming the text
- and secondary criteria, like the document's name.

The efficient usage of these three retrieval types and an appropriate combination of them should yield not only a fast response, but also an accurate and precise information.

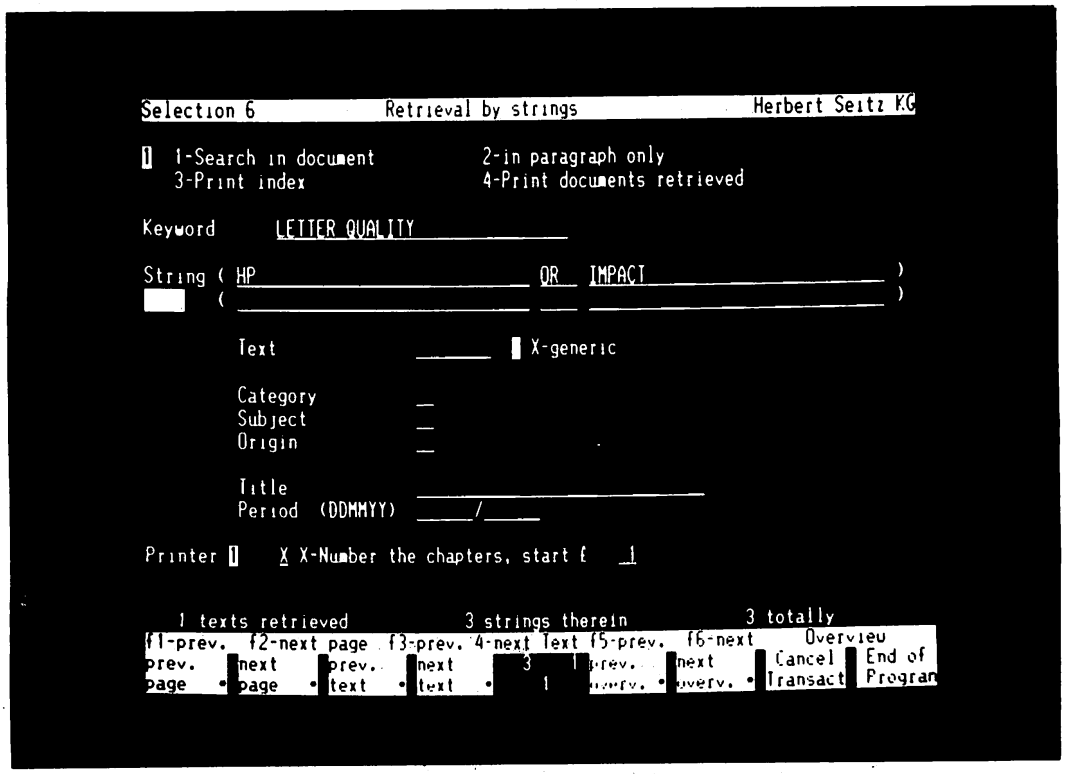
In the following paragraph we will discuss these three categories in more detail:

- Keywords:
  This means that you maintain an index - a thesaurus - of descriptors related to individual documents or parts of it. This approach gives the best response time, however, it requires the storage and maintenance of a keyword index. Anyhow, this approach is a must for extremely large text data bases - we will show later how index maintanance can be done at least semi-automatically.



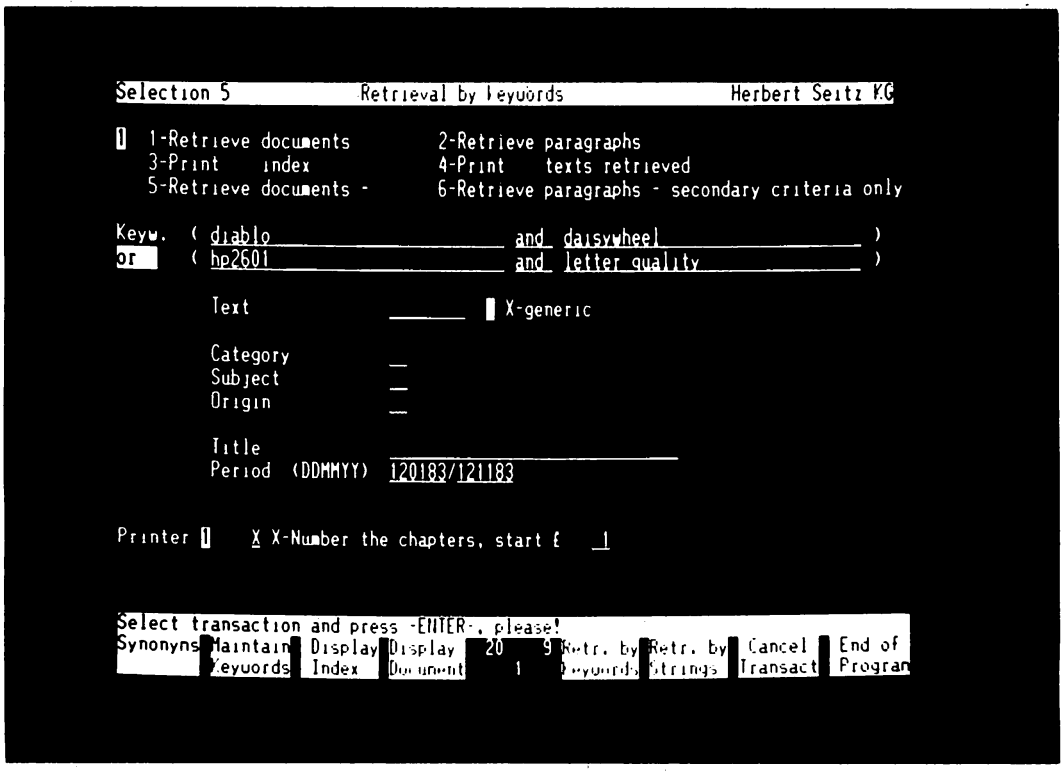
- Character string patterns:
  This approach is very flexible, as no predefined index is used. Every information within the text, every character string serves as information and thus as possible retrieval element.

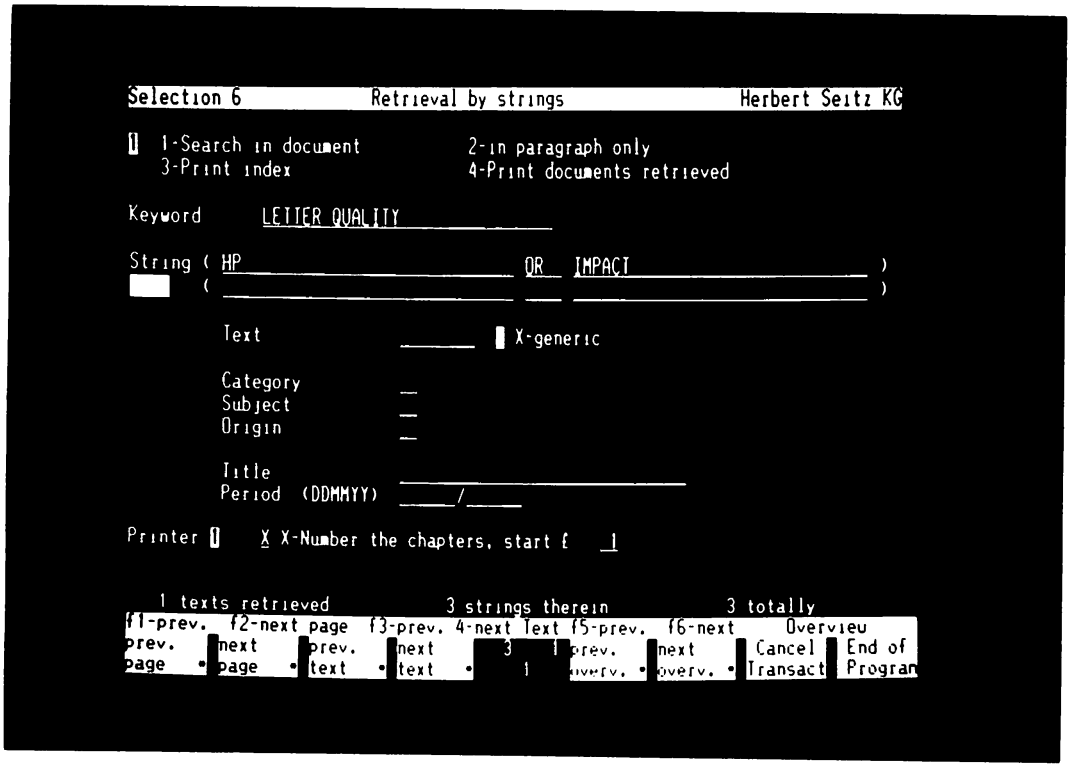
  The associated problem is speed. The IDT-EF currently is able to scan about 1000 to 3000 lines per minute (on a hp 3000-44). This includes reading text, upshifting, concatenating hyphenated words, searching and selecting.




- Secondary criteria:
  Every document, once it is created, has some criteria related to it such as the document name, creator, title, theme, date of creation. The IDT-EF maintains a number of additional secondary criteria such as titles, categories and so on. Secondary criteria usually are not random access criteria (except the document name), but they are a means of making retrieval more concise and faster, as they divide a large number of documents into a subset that can be substantially smaller. Especially we would like to mention a feature that we call "generic document names", a fast way of retrieving meaningful subsets.

With the IDT-EF you can use each of these retrieval methods separately, and you can combine the three methods at the same time, thus allowing a maximum of flexibility, accurateness and speed.



## Extended Search Methods

The first step is to search for an exact or at least for a generic match of a keyword or a combination of keywords.

However, we learnt very soon, that this does not fulfill all requirements.

There are at least two types of extended search methods desireable:

- retrieval by corresponding keywords and
- retrieval by synonyms.

## Retrieval by corresponding keywords

Searching in unstructured data provides some challenges. Some of these can be addressed, when you can also find corresponding keywords. Some examples:

- In Europe we find a great variety in writing foreign names: For example, we were told that there are 8 ways to spell the name GHADHAFI.

- Misspelled words are also phenomena you should be aware of, like MITERRAND or MITTERAND instead of MITTERRAND.

- And in some languages we find many different word forms. The German word for house - HAUS, also occurs as HAUSE HAUSES HÄUSER and HÄUSERN.

There are algorithms that are able to recognize the corresponding words based on a factor for correspondence.

## Retrieval by Synonyms

This means that retrieval covers a master-keyword and other keywords that have the same or a similar meaning, where in some cases a pattern match exists, in others not.

Some examples:

The above mentioned problem of spelling names is a case where some pattern correspondence exists.
But many times there is set of words, where this is not the case:

COMPUTER
DATA PROCESSING
DP
EDP

might be defined as synonyms.

In a multilingual environment you might define a keyword's translation as synonyms.
I.g. my country is called GERMANY here, but DEUTSCHLAND or ALEMANIA or TYSKLAND or ALLEMAGNE or ALEMANHA or GERMANIA in other parts of the world.

There are some problems associated with these techniques of extended searching:

- Synonyms must be defined before they can be used in document retrieval.

- Implementation is not just a trivial task.

- The biggest challenge, however, is performance: With the processors currently existing in the hp world (today is October 1983) response time might become a serious issue for three major reasons:

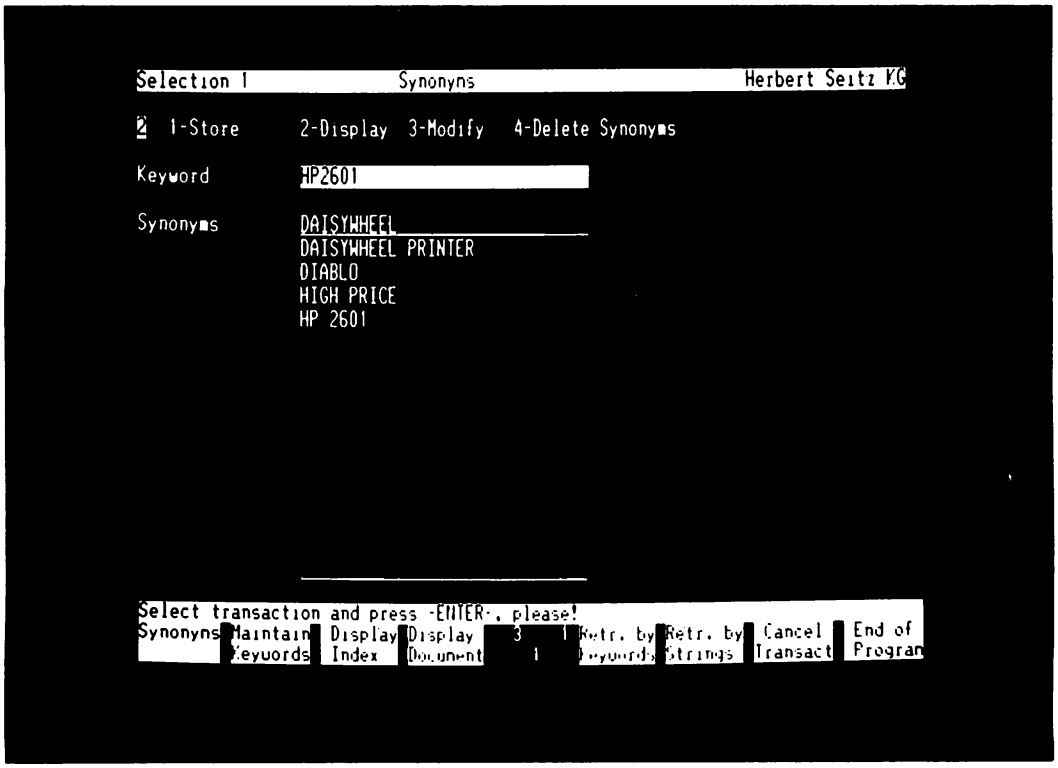
o some of the 3000 family's processors provide too little power for these non-trivial and non day-to-day data processing functions

o the number of disc accesses these systems can handle within a certain amount of time and the number of file-blocks that are present con-currently in main memory is limited

o mass storage systems with their mechanical access mechanisms are still relatively slow.

The tests we have done so far indicate that response times would be increasing in order of magnitude on today's computer systems if these techniques of enhanced searching would be fully implemented.

In order to overcome these shortcomings at least partially, we offer today online information about synonyms

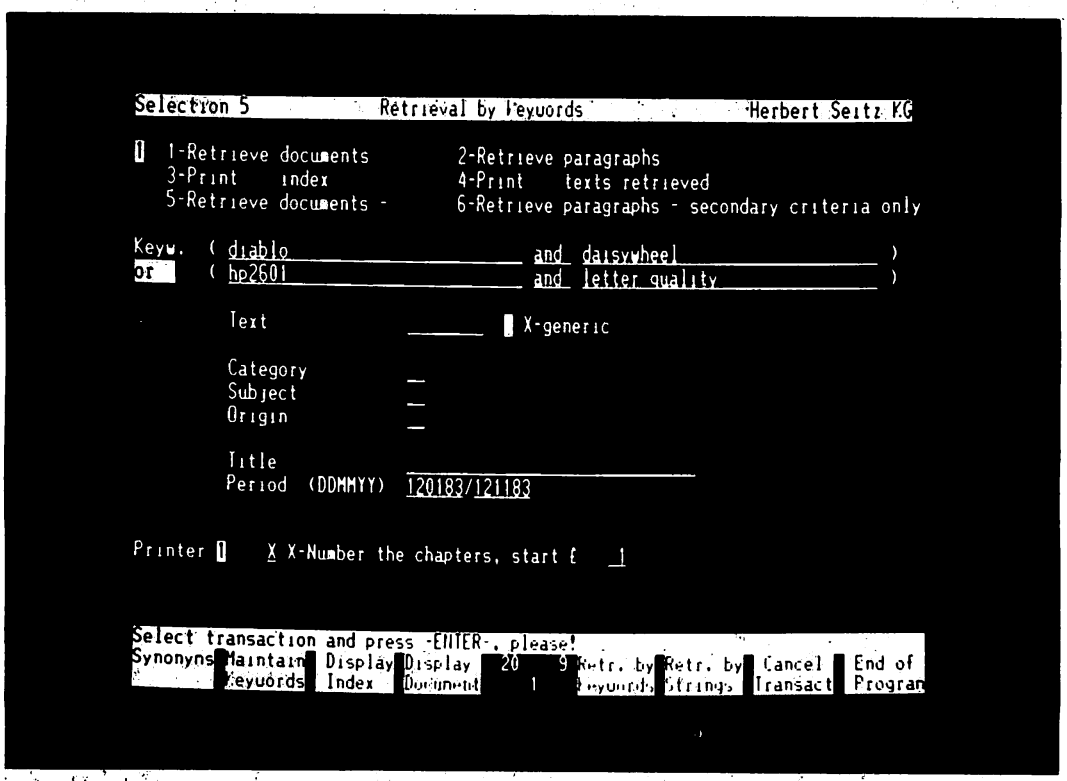



and similar keywords.

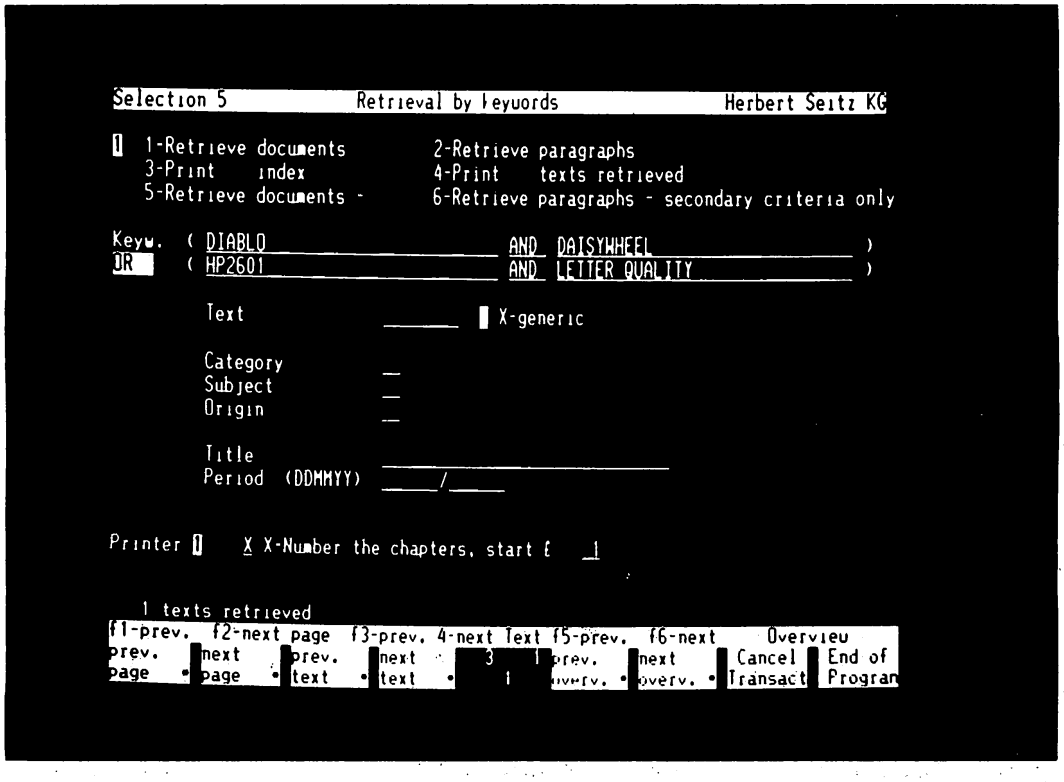
Reasonally efficient programming techniques for enhanced retrievals are already developed. We expect only a few more months (years?) waiting for faster mass storages and more powerful processors before these techniques can be made available in real life programs.

## The User Interface - Retrieving documents

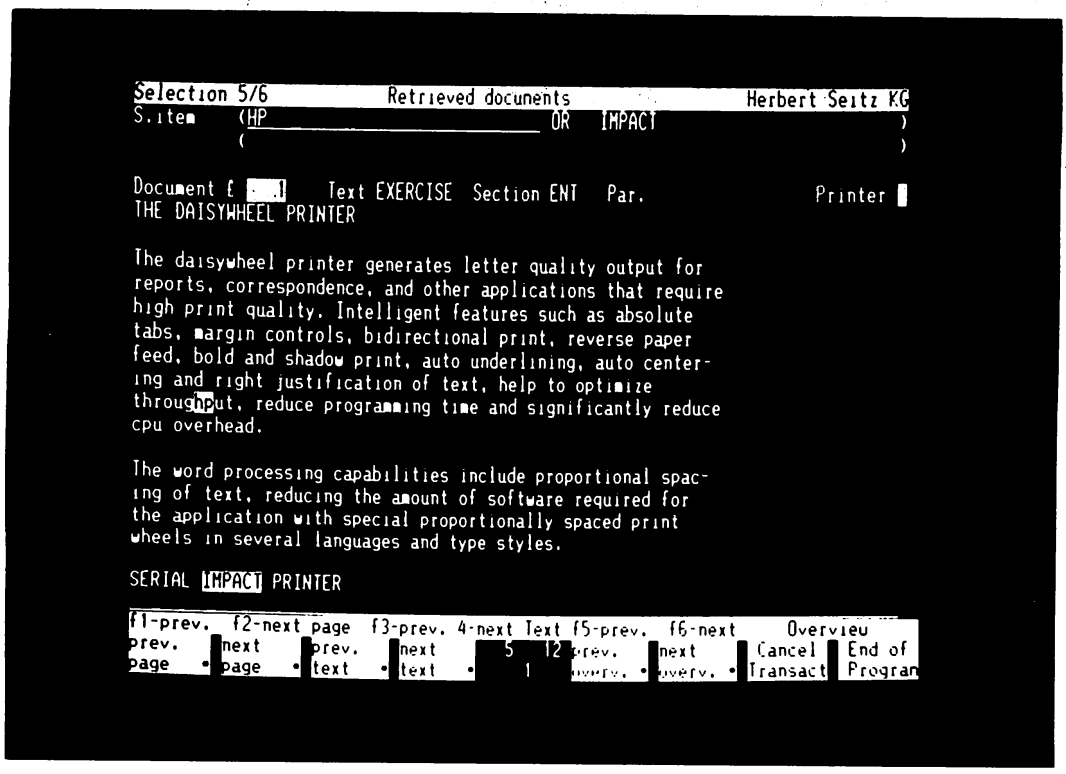
We selected a menudriven and softkey-oriented user interface. The user "fills in the blanks", all the information for a transaction at the same time, that is sent as a block to the computer. The user can define up to 5 primary keys plus secondary criteria. Primary keys are combined by the boolean AND / OR.




The number of retrieved documents and the number of retrieved strings are displayed together with the selection criteria.




Global function selection and fast browsing through the documents is controlled by softkeys, where no more than two levels of key labels exist.



Fast browsing thru documents means, that by pressing one function key, you should be able to display a list of retrieved documents. Just one more strike and you display the pages of a text. A handy print function should also be available. We designed an user interface which can be easily introduced to non-dp personnel. At our beta test site for instance, the system is used by journalists, that have very little practical experience with computers.

**The User Interface - Thesaurus management**

Of course, keyword index management should be done online only - and semi-automatically. That means it should be possible to define keywords with a minimum of work. We support two methods: first you can mark any word - or any pattern - within the text so the system proposes this pattern as a keyword. Second the system scans the text for all "words". In our system this is any combination of national alphabetic characters. Then it strips out all filler words of the current language in use, like in French LE LA ET OU and many more. After that, the words are displayed for manual modification and/or selection (each word only once - of course).

Again it is necessary - as for a word processor - to give full support to national character sets and languages.

By the way, it is an interesting side question how a "word" should be defined, if for example, a hyphen is part of a word or not. With current hp terminal hardware you run easily into trouble, because there exists no "Auxiliary Space" to define words as BUENOS AIRES and because only one type of hyphen exists where we would need two.

Of course, it must be possible not only to store, but also to modify, replace and delete keywords easily.

**The User Interface - Customization**

To make the system fit into a user environment, it is necessary to make it user customizable. Some of the features that should be configurable are

- Character sets
- Language used in documents
- Language in screens and messages
- Date format
- Printer assignment
- Text display format
- Sort criteria for retrieved documents.

The linguistic aspects are covered in more detail in "Edinburgh Proceedings, F.J. Boll - Linguistic aspects of word processing".

## Internal Structures - Searching and Sorting

The hp 3000 allows efficient string handling via certain firmware instructions or indirectly via SPL. We would be very unhappy if SPL one day would become obsolete, because it allows almost optimal algorithms for scanning, searching, moving bytes, upshifting and so on. Especially we would like to mention that in SPL you can almost directly use instructions like

```
SCAN BYTES,
COMPARE BYTES,
MOVE BYTES with upshift.
```

Programming for unstructured data applications like word and text processing becomes more efficient - and in a certain way even easier - when using pointers and stacks directly, where one needs a minimum of instructions and a minimum of code.

## Internal structures - Data Management

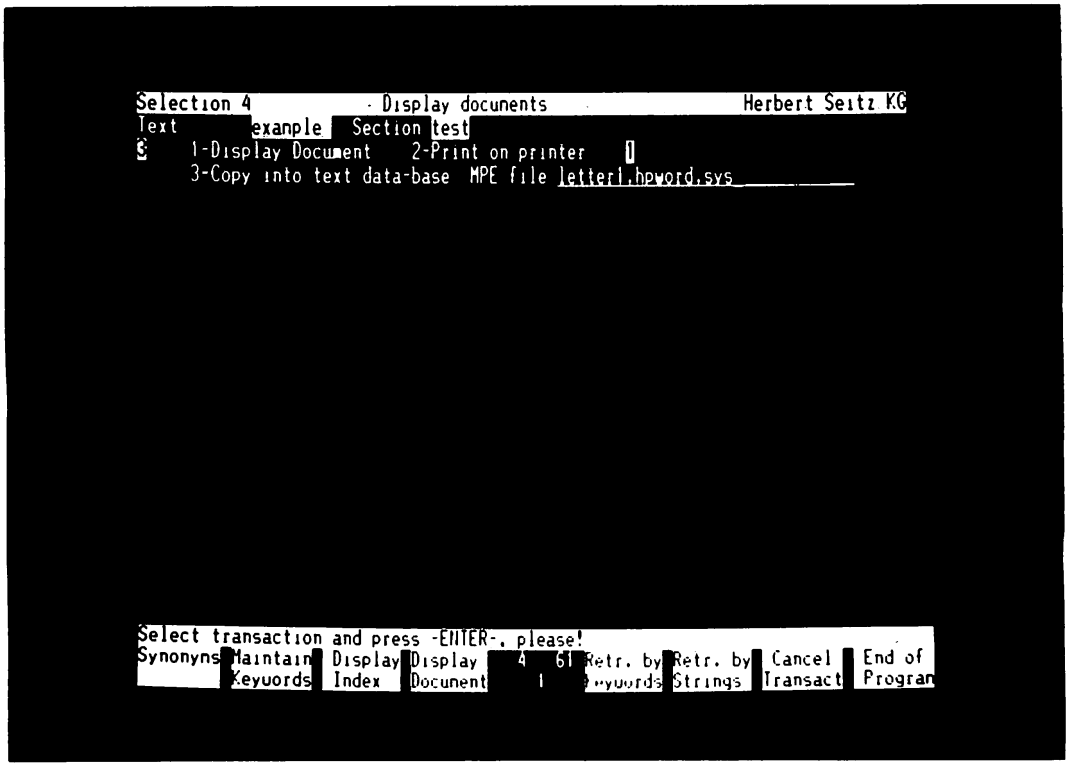
On user level you need a multi-level indexed data base for storing documents and also an indexed data base for the thesaurus.

As we wanted to focus our efforts on application problem solutions we decided to use an existing data management system. We realized, that KSAM has some of the structures we needed, inherently, IMAGE, however allows at least to emulate them against the user interface.

We realized that with both systems the interface might have had the same outlook, although the internal structures are so different.

And we choose IMAGE because we think that IMAGE is implemented in a much better way than KSAM, for example locking, the transaction monitor, access by a hp query language, and finally data integrity are items where IMAGE is by far better than KSAM. So we choose that file system although it meant some additional work to make it behave like an index sequential system.

In the beginning we designed our retrieval system as an enhancement to our integrated data- and word processing package **IDT**. We soon realized that we needed an Interface to MPE-Files in ASCII-Format. We wanted to be able to copy all kind of texts into our data base, like texts written by EDITOR, TDP, HP WORD or whatever. Thus this retrieval system can be used now by anybody having a hp 3000 computer, it is not restricted to a certain wordprocessor.The capability to re-convert printed documents (out of a library for example) into computer-processable ASCII format for our document data bases, is the next step in our plans.



## References

**Franz J. Boll**, Linguistic aspects of word processing, 1983, Edinburgh Proceedings

**Joachim Geffken**, User friendly applications in commercial realtime data processing, 1981, Orlando Proceedings

**Joachim Geffken** (36) is resident of Bremen, West Germany. He is cofounder and part owner of the Herbert Seitz KG, an international software supplier and hp OEM.. Joachim Geffken has studied law as well as computer science at the University of Hamburg. He is Vice Chairman of the German National Users Group and has served on the IUG Board of Directors. Off business hours he works as a publisher and author. He also is actively involved in offshore sailing.

**Franz-Josef Boll** was born in 1951 in Mühlheim near Frankfurt - West Germany. After high school, he studied in Frankfurt to become a teacher for mathematics and social science. After one year as a teacher, F.J. Boll joined the computer community as a programmer and system's analyst. After two years he started his own dp-development company and designed the **IDT** word processing system, the **IDT-EF** retrieval program. He also does linguistic research.

His hobbies are playing piano and church organ. F. J. Boll is also a frequent traveller through South America.