As a matter of fact, the search quality characteristics, such as recall and precision, should not be affected by the choice of known technical solutions used by creators of the IR system in realizing BSR. In other words, any implementation methods of comparison and selection are logically equivalent in the sense that a search will yield the same output. Therefore, in realizing the given processes, use is made of another indicator (for practical reasons this indicator is used only in realizing BSR), namely, the time spent by the computer to process them. Therefore, whatever choice of technical solutions is made, it is made primarily on the basis of this indicator. We say "primarily" because sometimes one has to take into account other considerations, for example, the availability of primary storage, in particular when the system is being created on a microcomputer basis. In any case, subsequently, when ranking different technical solutions, we will first consider the time needed for the search carried out in the BSR. We should emphasize that this chapter, although of interest to those directly involved in the creation of software implementing the BSR design, is intended mostly for those who take an interest in the creation of the IR system as a whole and who want to know the basic ideas underlying the creation of the BSR design.

8.2 Sequential File and Sequential Access

As is seen in Figure 4.8, a flow of documents is entered through the IR system's input (input 1). As the result of indexing, each newly input document has its presentation in the IRL (document profile). As described earlier, document profiles are necessary in a search process, and since document profiles are used in a search for all requests coming into the system, a need arises for their storage, a step realized in BSR. Yet any output of the systems is not just a set of retrieved (in response to a request, of course) document profiles; it is rather a collection of documents corresponding to these retrieved document profiles. This means that the input documents also have to be stored in the IR system.

There is a number of well known ways of storing information, which are referred to as methods of *file organization*. Each of the known methods has its own advantages and shortcomings having to do with the way of accessing the information stored in the file, or with an efficient usage of memory, or with the file's maintenance. Practically every file organization known in computer science provides a more successful solution to a certain class of tasks (applications): in our context a "more successful solution" means a more rapid solution, as we have noted. Now the question arises: Which file organization should be used in the IR system? There is no single answer to this question, and, for example, for the Boolean systems and SMART systems the answers are different. It should also be noted that the Boolean systems can use any file organization, including the one that is best for the SMART system, whereas the file organization that is the best for the Boolean systems is not usable in the SMART system. We will consider this distinction in more detail later on.

We start the analysis with the simplest file organization. Let us assume that 10,000 documents are to be used as the search collection of an IR system, which means that the system needs to store not only 10,000 original documents but also 10,000 document profiles. It is probably easiest to store the original document and its document profile together. In other words, for every document entered into the system, upon its indexing, a *record* can be formed (Figure 8.1) consisting of two fields, namely field 1 and field 2. Field 1 will contain the original document, and the document profile will be stored in field 2. For the 10,000 documents, naturally, there must be 10,000 such records; that is, a file consisting of 10,000 records will be set up. The next question arises: How are these records to be organized in the file and what kind of access method will be used, that is, what will the file structure look like? The two access methods are sequential access and random access. The former is simpler and its realization involves a sequential file (Grosshans, 1986). *A sequential file is a file in which a record is stored immediately following the previous record.* A sequential file is also known as a *serial sequential file*. Figure 8.2 illustrates a sequential file consisting of 10,000 records.

In sequential file organization, the records are inserted into the file in order of their arrival, that is, in chronological order. In other words, a sequential file is organized by adding or *appending* records only at the end of the file. Thus, the first record in the file is the oldest and the last record in the file is the one most recently added (the last addition). The only access method that is valid for a sequential file is sequential access. *Sequential access is access that takes records in order, looking at the first, then the next, and so on.* Therefore, a subject search in the sequential file (Figure 8.2) can be performed as follows. After reading the first record of the formed file, the search program compares the document profile (the content of field 2) with subrequests of the query formulation constructed from the search request entered into the system. Whenever any subrequest is

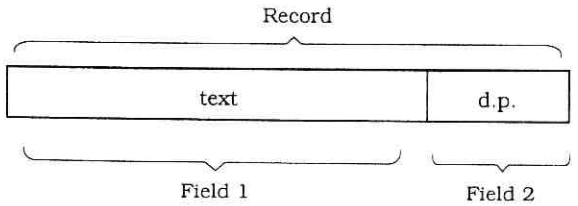

Figure 8.1
Document record.