On Some Problems of Online Data Base Systems

RYOSUKE HOTAKA*

Abstract

Several problems on Online Data Base Systems are discussed based on the actual experimental system BSCL 1.

Among interests of this paper are:

(1) Implementation of the garbage collection of main storage using the technique of compacting.

(2) Concurrent information storage and retrieval of online data base experimented on the system BSCL 1.

(3) Implementation of a special inverted file.

(4) Introduction of system files to describe the data base.

(5) Tests of online terminals by the terminal simulator.

(6) Some investigations to the error recovery of the online database system.

(7) Some investigations of resource allocation problems peculiar in online data base system.

1. *History*

Nippon Software Co. began a project on online IR system in 1969.

At first, the project aimed the definition of a new language suitable for online IR. The work continued about half a year and defined SCL 0.5 (Symbolic Command Language).

SCL 0.5 is a self-contained system. In order to satisfy various necessities, it must have extensively large specification.

Thus when we began to implement the system, we had to restrict the specification considerably. It is BSCL 1 a subset of SCL 0.5 that we finished in the first stage.

In implementing an online system, there are as many problems in the interface between operating system and BSCL 1 as in the specification of the language.

2. *The Object of the Online Data Base System*

First, we give the definition of the data base:

Definition: The data base is a set of filed information interrelated each other. The access and the maintenance of files in the data base is done preserving

This paper first appeared in Japanese in Joho-Shori (Journal of the Information Processing Society of Japan), Vol. 12, No. 12 (1971), pp. 738~745.

* Nippon Software Co., Ltd.

integrity. Further the information stored in the file has the compatibility between them.

ODB (online data base) is a data base the access of which is made possible from the online terminals.

The object of ODBS is to support the following services in the online environment.

(1) Standard method of accessing data base. This facilitates the integrated maintenance of ODB, and makes the user program immune to the minor changes of physical configurations or the operating system.

(2) Data common to several application program occupies only one physical place, thus making the compatible maintenance and updates easy.

(3) The compatibility of batch and online processing.

(4) Standard and extensive support for the error recovery.

(5) Reduction of developing cost of user program, required memory storage and processing time.

3. *BSCL 1 System*

3.1 *The Overview of the System*

The machine used to implement BSCL 1 is FACOM 230-60. Four terminals run concurrently. Each of them corresponds to a task. The master task generates and supervises these four tasks, and it can stop and restart each one of them. These five tasks operate under the usual operating system. (Fig. 1)

Two tasks corresponding to terminals are realized as terminal simulators.

The master task accepts commands from the operator's console and responds the answer on it.

The user can input programs written in BSCL 1 language from his terminal

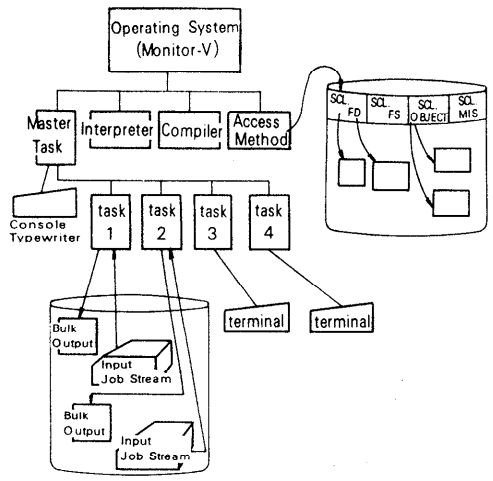

Fig. 1 General construction of BSCL 1

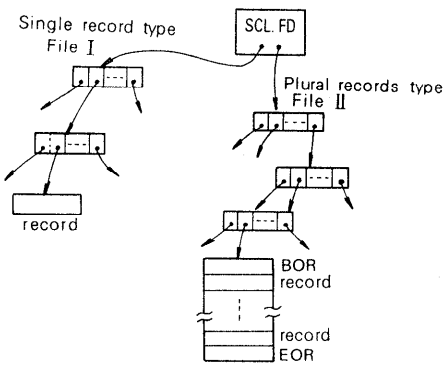

Fig. 2 File structure of BSCL 1

and execute them.

The program can be registered after compilation and shared between terminals. Therefore the user program as well as the system program is made reentrant.

The system occupies 45 k words (1 word=36 bits), but could have been made smaller if proper optimization had been attempted.

The terminal simulator is, in fact, a combination of two sequential files, one for the input and one for the output.

3.2 *Task Management*

BSCL1 runs as one job under the control of MONITOR-V (one of the operating systems of FACOM 230-60). Since the master task supervises other four tasks, some degree of task management is necessary.

(1) Each programs must be made reentrant and run concurrently.

(2) Proper exclusive control must be done in sharing the data base.

(3) The master task must be able to stop all terminals synchronously and restart them later. These facilities are used in garbage collection (See 3.3).

These points will be discussed in detail.

(1) is realized using main storage management routine stated in 3.3 by allocating main storage to each task as it is required.

We realized (2) by distinguishing two modes of control to use various resources belonging to the data base.

The first mode, "write" mode, admits only one task to do some operation on the specified resource of ODB. This mode is particularly useful when one task attempts to write some part of ODB. In that case, every task must seize the resource in "write" mode.

The other mode, "read" mode, permits more than one tasks to do some operations on the specified resource of ODB.

Similarly every task that attempts to read some part of ODB must seize that resource in "read" mode.

Though MONITOR-V has only one system macro which is convenient to implement "write" mode, we can implement "read" mode using this macro. The implementation of (3) is not so easy. We adopt the following procedures:

(i) The command which orders all the terminals to stop is accepted by the console typewriter, the master task issuing the direction.

(ii) Each task corresponding to a terminal checks periodically if the "stop" direction has been issued by the master task.

(iii) If a task detects the "stop" signal, it returns "O.K." signal to the master task and stops.

The above procedure does not work if a task has been waiting when "stop" direction is issued. To modify this, the next procedure is necessary.

(iv) The master task checks if all of the tasks have stopped, and makes waiting tasks reply.

3.3 *Main Storage Management*

BSCL 1 permits each of its terminal job to get variable amount of main storage dynamically. The function and feature of the management is as follows:

(1) BSCL 1 gets fixed size of main storage from MONITOR-V.

(2) After that BSCL 1 gives each terminal job main storage of variable length as it is requested.

(3) The terminal job returns main storage after the use of it.

(4) BSCL 1 manages the control of main storage by the first-fit method (see [2]), and performs garbage collection with compacting.

While garbage collection of main storage is being performed, every task must be halted.

This garbage collection with compacting takes about 2.6 seconds in the most difficult case (collecting 50 words×163 high speed core memory plus 50 words ×655 low speed core memory).

3.4 *Data Management*

A new file structure and its access method (see Fig. 2) is designed, but its implementation was partly unfinished.

This file structure is so to speak a combination of an indexed sequential file and a partitioned file.

It has a directory of unlimited number of levels, and its member can contain unlimited number of sequential records.

The addition, creation, deletion and the modification of the file is admitted without any limit.

But this kind of file structure and its access method is excessively difficult to implement and we could not complete it in the prescribed time limit.

The following problems arise:

(1) The ability to add and update the file without any limit makes the management of the directory, efficient use of both space and time very difficult.

We find quick retrieval and online updating using hierarchical directory almost inconsistent each other.

(2) When we use directory, even retrieval of data necessitates the exclusive control of lower level directory, thus degrading the efficiency of retrieval.

(3) Efficient sharing of data base and security keeping turn out to be two difficult and incompatible requirements.

3.5 *System File*

BSCL 1 has four system files which hold the management information of ODB.

These all four files have the file structures explained in 3.4.

(1) SCL. FS keeps all the definition of user's file structure.

(2) SCL. FD keeps all the information about user's files except the information contained in SCL. FS.

(3) SCL. OBJECT keeps registered object programs made by BSCL 1 compiler.

(4) SCL. MIS keeps the total information about the data base, e.g. mappings of logical files to physical location.

3.6 *Language*

Though the language of BSCL 1 is a subset of SCL 0.5, slight modification is done in reality.

Moreover several defects of the language specification appear as we type in from the terminal. Example of the language is shown in Fig. 3.

```
*** S C L 0.5 ...  START                                   TERM. NO. = 3
<SCL0.5>
0001
0002  (PROC)EISFILE
0003  DCL   FPA INTE*2(FLNAME CHAR(12),RES CHAR(116),RTCODE,FLAG,DILEN,
0004  KEY CHAR(20));
0005  MOVE 'EISFILE      ' TO FLNAME
0006  MOVE 0 TO FLAG
0007  OPEN INPUT FPA=FPA
0008  IN: TYPE IN KEY'
0009        (KEY='12345678901234567890', GOTO END)
0010  IN FPA=FPA
0011  TYPE OUT EISFILE_ENG
0012  TYPE OUT EISFILE_FREN
0013  TYPE OUT EISFILE_GERM
0014  GOTO IN
0015  END: END
*** S C L 0.5 ...  END                                     TERM. NO. = 3
```

2nd line	(PROC)	declaration of the program
3rd line	DCL	declaration of variables
5th line	MOVE	a command to move data
7th line	OPEN	a command to open a file
8th line	TYPE IN	a command to receive characters from a terminal
9th line	(c, s)	a conditional statement, to execute the statement s if the condition c is true
10th line	IN	a command to read data from a file
11th line	TYPE OUT	a command to send characters to a terminal
14th line	GOTO	a command to transfer control
15th line	END	declaration of the end of the program

Fig. 3 An example and the explanation of a sample BSCL 1 program

4. *The Problem of ODBS*

In the process of developing BSCL 1, we notice the following difficult problems which must be solved in the generalized ODBS.

4.1 *Deadlocks*

This is not the particular problem appeared in ODBS, but the problem of multiprogramming. Yet ODBS must observe this problem more nervously, for ODBS encounters more complex ways of assigning and sharing resources in ODB.

More complex algorithms to avoid deadlocks without sacrificing the efficiency, or recovery procedures from deadlocks must be investigated.

An algorithm preventing from deadlocks is given by the author in [1].

4.2 *Recovery from Errors*

Very complex and expensive error recovery procedures will be necessary if online updating is admitted.

The reason that makes error recovery of online updating difficult is that ODB changes from time to time as updating transaction or request comes in, and the time when the error occurs is completely unpredicable.

We think the following preparations are necessary to recover an online updated data base.

(1) Periodical and total copy of the data base.

(2) Event logging and taking journal. It is necessary to take before and after image of each changed data of ODB.

(3) Standardization of file structure. This is necessary to identify various information of ODB and diminish the load of error recovery procedures.

4.3 *Resource Allocation*

In ODBS, more delicate way of resource allocation is necessary.

First, there are several units of resource allocation, e.g. a user want to use particular record.

Secondly, there is a variety of time span during which the user want to use resources. The ordinary operating system does not have the ability to handle this variety.

Third, there are different kind of modes to use resources. In BSCL 1, we distinguished only two modes i.e. "write" and "read" modes. But a user might want to use a resource in "read while changing" mode

i.e. He doesn't mind the other user updates the data he is reading.

"protect against current updates" mode

i.e. He doesn't mind the other user is reading the data as long as other user doesn't updates.

These three independent ways of using resources will probably require complex control of ODBS.

5. *Acknowledgement*

The author is very grateful to Messrs M. Takagi, M. Kobayashi, K. Hatoya and K. Morita for their kind help.

References

[1] Ryosuke Hotaka: On a method of detecting a deadlock, appeared in this issue.

[2] D. E. Knuth: The art of computer programming, Vol. 1, Fundamental algorithms, Addition-Wesley, 1968.

[3] CODASYL Systems Committee, Technical Report: "Feature analysis of generalized data base management systems". May, 1971.