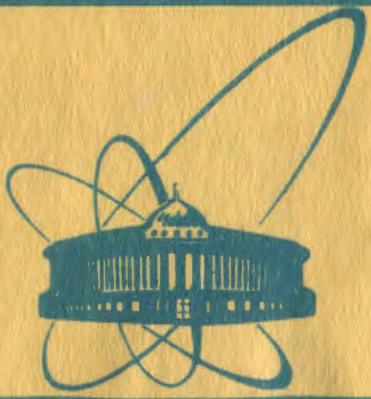

сообщения
объединенного
института
ядерных
исследований
дубна

Р10-80-861

Л.С.Нефедьева, А.И.Салтыков

ДИРЕКТИВЫ СИСТЕМЫ ОБРАБОТКИ СПЕКТРОВ (СОС)

1980

I. Введение

Система обработки спектров на БЭСМ-6, или сокращенно система СОС, представляет собой файловую систему обработки данных [1].

Файл [2,3] – это массив данных, имеющий наименование и расположенный на одном из внешних носителей информации (МЛ, МД, МБ или перфокартах). Файл состоит из отдельных записей (рекордов). Запись – это элементарная единица информации, хранимая на внешних носителях.

В целях предоставления пользователям удобного аппарата общения с системой СОС разработан специальный язык директив [5], который для краткости будет называться языком СОС.

На языке СОС пишется задание на обработку информации, которое является главной программой пакета задачи [4]. Задание охватывает ту часть процесса обработки информации, которая происходит на уровне файлов и записей. В задании сосредоточена вся информация о файлах, используемых в пакете задачи.

Из задания можно обращаться к программным модулям, которые производят обработку информации уже на уровне оперативной памяти. При необходимости программный модуль может производить считывание и запись файлов посредством системных программ CGET и CPUT . Однако имена файлов программный модуль может получить только из задания и притом только в качестве фактических параметров.

2. Структура языка СОС

Язык СОС имеет фортраноподобную структуру. Запись задания на языке СОС состоит из отдельных операторов, каждый из которых начинается с ключевого слова и располагается на новой перфокарте.

Аналогично языку ФОРТРАН в языке СОС имеются декларативные и выполняемые операторы. Декларативные операторы описывают уже созданные файлы и вновь создаваемые файлы. Выполняемые операторы реализуют операции над файлами и записями: перепись информации из одних файлов или записей в другие, исключение файлов и записей и переименование файлов. К выполняемым операторам относятся также оператор вызова программных модулей и оператор цикла.

Правила записи операторов языка СОС во многом аналогичны правилам записи операторов ФОРТРАНа. Текст оператора располагается в колонках 7-72, шестая колонка предназначена для указания признака продолжения, в колонках 1-5 можно указать метку оператора. Колонки 73-80 отводятся под комментарий. Пробелы в записи операторов (за исключением записи текстовых констант) игнорируются.

Для обозначения различных объектов, используемых в задании (файлов, программных модулей, магнитных лент и т.д.), служат идентификаторы. Понятие идентификатора в языке СОС совпадает с принятым в языке ФОРТРАН. Однако вводится ограничение: идентификатор не может начинаться с буквы С. Исключение составляют имена магнитных лент, для которых допускаются любые идентификаторы.

3. Операторы PROGRAM и END

Задание начинается с заголовка, который указывается посредством одного из операторов:

```
PROGRAM  <идентификатор задания>
PROGRAM  <идентификатор задания>  (1)
PROGRAM  <идентификатор задания>  (2) .
```

Цифры 1 и 2, которые могут быть указаны в скобках после идентификатора задания, определяют режим работы системы:

1 - система работает с большим буфером (5К); 2 - с малым буфером (1К); если режим не указан, то с малым буфером.

Буфер служит временным хранилищем информации при выполнении операций обмена между файлами и записями. Использование большого буфера ускоряет указанные операции обмена. Однако в этом случае для программных модулей остается меньше места в оперативной памяти.

Оператор END является последним оператором задания.

4. Декларативные операторы

К декларативным операторам относятся: оператор описания файлов FILE , оператор создания новых файлов CREATE , а также фортраноподобные операторы COMMON и EXTERNAL .

Все файлы, используемые в задании, должны быть описаны.

Декларативные операторы располагаются в начале задания. Порядок их взаимного расположения, как правило, не существен.

4.1. Оператор описания файлов

Описание файлов, уже созданных до выполнения задания, производится посредством одного из следующих операторов:

```
FILE  <список> ( INPUT  , <имя ленты> )
FILE  <список> ( OUTPUT , <имя ленты> )
FILE  <список> ( UPDATE , <имя ленты> ) .
```

Список содержит имена файлов, выраженные идентификаторами и разделенные запятыми. Имя ленты – это наименование ленты, указанное в карте заказа ресурсов [4].

Примеры:

```
FILE T16, ABCD, RZ(INPUT,LN F3)
FILE ADR, ALFAPR (OUTPUT,TAPE1).
```

Каждый файл может иметь один из типов: INPUT , OUTPUT или UPDATE . Тип файла определяет множество операций, которые разрешается выполнять над этим файлом. Такими допустимыми операциями являются:

1) чтение (для типов INPUT и UPDATE);

2) запись с затиранием уже имеющихся рекордов (для типа UPDATE);

3) запись с расширением, т.е. с добавлением новых рекордов (для всех типов).

Тип файла имеет силу только в пределах данного задания.

Правила:

1. Каждый файл должен быть описан и притом только один раз.
2. В одном задании может быть описано не более 20 файлов, расположенных на магнитных лентах. Общее количество записей, содержащихся в этих файлах, не должно превосходить 100.

3. Операторы FILE должны предшествовать операторам CREATE, использующим соответствующие файлы или записи для создания новых файлов.

4.2. Оператор создания новых файлов

Файлы, вновь создаваемые в задании, описываются следующим операторами:

CREATE	<имя файла> (INPUT , <имя ленты>) = <список>	(I)
CREATE	<имя файла> (UPDATE , <имя ленты>) = <список>	(2)
CREATE	<имя файла> (TIME , <имя ленты>) = <список>	(3)
CREATE	<имя файла> (TIME) = <список>	(4)
CREATE	<имя файла> (OUTPUT , <имя ленты>)	(5)
CREATE	<список> (WORK)	(6)
CREATE	<список> (WORK ,<длина>) .	(7)

В операторах (I)-(4) список после знака = содержит имена уже имеющихся в системе файлов и записей, из которых создается новый файл, имя которого указано слева от знака =. Операторы (5)-(7) соответствуют вновь создаваемым файлам, которых к началу выполнения задания в системе еще нет.

Файлы типов INPUT, OUTPUT и UPDATE,полученные путем объединения уже существующих файлов и записей, являются постоянными файлами и могут быть использованы в последующих заданиях.

Файлы типа TIME создаются лишь на время выполнения данного задания. В случае, когда вновь создаваемый файл типа TIME составляется из файлов и записей, расположенных на различных магнитных лентах, имя ленты для него не указывается. Файлы типа TIME допускают любые режимы использования (аналогично файлам типа UPDATE, описанным в п.4.I).

Вновь создаваемые файлы (5) типа OUTPUT перед выполнением задания регистрируются лишь во временном каталоге.Если при выполнении задания в такой файл производится запись, то по окончании выполнения задания этот файл регистрируется в паспорте соответствующей магнитной ленты. Если же в процессе выполнения задания в такой файл не произведено ни одной записи, то в паспорте магнитной ленты он не будет зарегистрирован.

Файлы типа WORK являются рабочими файлами. Они хранятся на МБ. Файлы (6) называются рабочими файлами с произвольным доступом.Они используются в задании по тем же правилам,что

и файлы типа UPDATE (см.п.4.I). Файлы (7) являются рабочими файлами с последовательным доступом. Обмен информацией с этими файлами производится по особым правилам.

В случае, когда новый файл типа INPUT, OUTPUT или UPDATE создается из уже имеющихся файлов, происходит их переименование. Если же новый файл создается из записей, входящих в состав уже существующего файла, то эти записи исключаются из "старого" файла и в нем производится перенумерация оставшихся записей. Во вновь создаваемом файле записи нумеруются в порядке их следования в списке соответствующего оператора CREATE.

Пример:

```
FILE A5,B8,R(INPUT,T)
CREATE FA(UPDATE,LSN) = A5(3),B8,AA(6),A5(1) .
```

Написанные операторы означают, что из файла A5 будут исключены записи A5(I) и A5(3), а из файла AA - запись AA(6).

Файл B8 теперь уже не может быть использован под этим именем.

Пусть в файле B8 содержится 4 записи. Тогда вновь созданный файл FA будет содержать 7 записей: FA(1), FA(2), FA(3), FA(4), FA(5), FA(6), и FA(7), которые соответствуют записям A5(3), B8(I), B8(2), B8(3), B8(4), AA(6), A5(I).

Если вновь создаваемый файл имеет тип TIME, то прежние файлы и записи, из которых он составляется, сохраняются. Однако, если некоторый файл использован для создания нескольких новых файлов (т.е. указан в правой части нескольких операторов CREATE), то он может быть использован под прежним именем лишь при условии, что все эти новые файлы имеют тип TIME.

Например, оператор

```
CREATE FT(TIME,T) = B8(2),R,A5(4),
```

написанный после двух операторов, указанных в предыдущем примере, означает, что, скажем, запись FT(4) можно использовать и под именем R(3) . Однако, если файл R будет указан в правой части хотя бы одного оператора CREATE, описывающего новый файл не типа TIME, то файл R уже нельзя использовать под этим именем. Запись B8(2) под этим именем уже не существует, поскольку файл B8 использован для создания постоянного файла FA.

Правила:

1. Вновь создаваемые файлы не могут быть описаны оператором FILE.
2. В файлах типа TIME нельзя увеличивать число записей в процессе выполнения задания.
3. Рабочие файлы, в которые не произведено ни одной записи, нельзя использовать в режиме чтения. При попытке чтения из еще "пустого" файла выдается соответствующая диагностика и выполнение задания прекращается.
4. Файлы, указанные в правой части операторов CREATE , не могут находиться в левой части операторов CREATE.

4.3. Оператор описания общего блока

Для описания общих блоков служит оператор COMMON , имеющий более простую структуру по сравнению с языком ФОРТРАН. Этот оператор может быть записан в одной из двух следующих форм:

COMMON / <имя блока> / <идентификатор> ,

COMMON / <имя блока> / <идентификатор> (<целое без знака>).

Здесь *имя блока* может быть любым идентификатором (в том числе и начинающимся с буквы C). Идентификатор, указанный после имени блока, не должен совпадать с именем файла или программного модуля.

Среди COMMON-блоков особую роль играет блок с именем COC1, который необходим для работы системы COC. Если этот блок не описан в задании, то при трансляции с языка COC на ФОРТРАН генерируется оператор

COMMON/COC1/CCCCCC (4200).

Если блок с именем COC1 описан в задании и имеет длину менее 4200, то устанавливается его длина 4200. Если этот блок имеет длину не менее 4200, то эта длина сохраняется после трансляции.

4.4. Оператор описания программных модулей

Если имя программного модуля является фактическим параметром в операторе CALL , то он должен быть описан в операторе EXTERNAL . Этот оператор записывается в той же форме, что и на ФОРТРАНе.

5. Выполняемые операторы

К выполняемым операторам относятся:
оператор обмена записями WRITE;
оператор переименования файлов RENAME;
оператор исключения файлов и записей DELETE;
оператор цикла DO;
оператор вызова системных модулей CALL.

5.1. Оператор обмена записями WRITE

Обмен информацией на уровне файлов и записей производится посредством операторов:

WRITE < список > .INTO. < имя файла > ,
WRITE < список > .INTO. < имя записи > .

Здесь список содержит имена файлов и записей, из которых переписывается информация. Если в списке файлов и записей указывается файл, то передается весь файл, т.е. все его записи. Если справа от .INTO.указано имя файла, то информация переписывается в режиме добавления новых записей к уже имеющимся в этом файле. Если же справа указана запись, то перепись информации производится с "затиранием" уже имеющихся записей, начиная с той, которая указана справа от .INTO.

Правила:

1. В списке файлов и записей слева от .INTO. не может быть указан файл типа OUTPUT или запись, относящаяся к файлу типа OUTPUT.

2.Если справа от .INTO. указана запись, то она не может принадлежать файлу типа INPUT.

3. В операторе WRITE нельзя указывать файлы с последовательным доступом (см. п.4.2).

5.2. Оператор переименования файлов RENAME

Переименование файлов в процессе выполнения задания может производиться одним из следующих операторов:

RENAME < список > .BY. < список > ,
RENAME <имя файла>.BY. < список > .

Список слева содержит имена файлов, подлежащих переименованию, а список справа - их новые имена. В первом случае каждому элементу списка слева должен соответствовать элемент списка справа. Во втором случае каждая запись из файла, указанного слева от .BY. , получает новое наименование согласно списку имен, указанному справа.

Пример:

```
RENAME A,B,BC .BY.AA,BB,BCC.
```

Здесь производится переименование файлов A,B,BC, в AA,BB,BCC соответственно.

Правила:

1. Можно переименовывать любые файлы независимо от вида и типа.

2. Длина списка, указанного справа от .BY. , должна быть равна длине списка, указанного слева от .BY . В случае, когда переименовываются записи, число имен в списке, указанном справа, должно быть равно количеству записей в файле, имя которого указано слева от .BY.

3. Все файлы, указанные в списке слева от .BY. , должны быть описаны операторами FILE или CREATE.

4. Все файлы, указанные справа от .BY. , не могут быть описаны операторами FILE и CREATE.

5. Новые имена файлов не должны совпадать с именами других файлов, используемых в данном задании.

6. Оператор RENAME нельзя использовать в цикле.

7. После выполнения оператора RENAME все переименованные в нем файлы можно использовать только под новыми именами.

5.3. Оператор исключения файлов и записей DELETE

Данный оператор записывается в виде

DELETE < список файлов и записей > .

Оператор DELETE исключает перечисленные в его списке имена файлов и записей из паспортов соответствующих магнитных лент. После выполнения оператора DELETE уже нельзя использовать исключенные файлы. Если же из файла исключены лишь отдельные

записи (так что в файле осталась хотя бы одна запись), то производится "вычеркивание" исключенных записей и перенумерация оставшихся записей.

Пример:

```
DELETE X1,X2,AB(8),R(3).
```

После выполнения этого оператора будут исключены файлы X1 и X2, а в файлах AB и R произойдет исключение указанных записей и перенумерация оставшихся записей.

Правила:

1. Оператор DELETE можно использовать только для постоянных файлов.

2. Список файлов и записей не может содержать более 15 элементов.

3. Все исключаемые файлы должны быть описаны операторами FILE или CREATE.

4. Нельзя исключить файл, который был вновь создан оператором CREATE вида (5)(п.4.2).

5. Нельзя использовать оператор DELETE в цикле.

5.4. Оператор цикла DO

Оператор аналогичен фортранному оператору DO, но имеет ограничение: в качестве начального и конечного значений переменной цикла и шага ее изменения можно задавать лишь целые константы без знака.

Примеры:

```
DO 5  I = 1, 50, 3
DO 18 K = 5, 12 .
```

Ограничением на использование оператора цикла является также запрещение вложения циклов.

5.5. Оператор вызова программных модулей CALL

Этот оператор аналогичен фортранному оператору CALL и может быть представлен в двух видах:

CALL <имя программного модуля>,

CALL <имя программного модуля>(<список фактических параметров>).

В качестве фактического параметра может задаваться:

1) целая константа в смысле ФОРТРАНа;
2) вещественная константа в смысле ФОРТРАНа;
3) текстовая константа в виде "< текст не более 6 символов >" ;
4) имя файла (идентификатор);
5) запись, т.е. имя файла и номер записи, указанный в скобках;
6) имя программного модуля, описанного в операторе EXTERNAL.

В качестве номера записи может быть задана целая константа. В случае, когда оператор CALL располагается внутри цикла, номер записи может быть задан как индексная переменная, совпадающая с переменной цикла.

Примеры:

```
      CALL BIT
      CALL ARGUM(F5,RR(2),5,2.E6, 'ABC')
      DO 12 I=1,6
   12 CALL RTF (RR(I),M18, 'C14O').
```

Пример задания:

Переписать содержимое файлов A1 и A2, расположенных на МЛ с именем FT1 , в файлы B1 и B2, вновь созданные на МЛ с именем PRIN . Переписать записи 9-13 из файла A3 на МЛ T5 в файл RB тоже на МЛ T5 в режиме дополнения к уже имеющимся там записям.

```
      PROGRAM  FCOPY
      FILE A1,A2(INPUT,FT1)
      CREATE B1,B2 (OUTPUT,PRIN)
      FILE A3,RB (INPUT,T5)
      COMMON/COC1/A(4096)
      WRITE A1.INTO.B1
      WRITE A2.INTO.B2
      DO 5 J=9,13
    5 WRITE A3(J).INTO.RB
      END .
```

Авторы благодарят Х.Кэнига и Т.С.Рериx за участие в обсуждении проекта языка СОС.

ЛИТЕРАТУРА

1. Нефедьева Л.С. ОИЯИ, Д10-7707, Дубна, 1974, с.476–480.
2. Кэниг Х., Нефедьева Л.С. ОИЯИ, 10-8556, Дубна, 1975.
3. Лефковиц Д. Структуры информационных массивов оперативных систем /перевод с англ./. М., "Энергия", 1974.
4. Мазный Г.Л. Программирование на БЭСМ-6 в системе "Дубна". "Наука", М., 1978.
5. Говорун Н.Н., Иванченко И.М., Нефедьева Л.С. Диалог в системах автоматизированной обработки данных. Киев, "Наукова думка", 1974.

Рукопись поступила в издательский отдел
26 декабря 1980 года.