В связи с размещением на сайте журнала (www.lib-journal.ru) демонстрационной версии автоматизированной картотеки книгообеспеченности (подробное описание картотеки изложено во 2-ом номере журнала в статье Б.П. Бочарова «Автоматизированная картотека обеспеченности учебной литературой») в редакцию поступают вопросы о возможности импортирования баз данных из электронного каталога в картотеку.

Отвечая нашим читателей, приводим материалы, содержащие методику разработки конвертора. Используя настоящую статью, а также размещенные на сайте журнала инструкцию и руководства, программисты библиотеки могут самостоятельно написать необходимую программу, как это было сделано в научной библиотеке Новосибирского государственного технического университета.

БОЧАРОВ Б.П.

ВОЕВОДИНА М.Ю.

AWK - универсальная программа работы с текстовыми файлами

Ни одна предметная область, в том числе и библиотечное дело, не может быть автоматизирована на сто процентов. Какие-то операции неизбежно продолжают выполняться в «ручном» варианте, для других по мере изменений, происходящих с объектом автоматизации, все равно приходится составлять дополнительные программы. Случается, что даже очень простая задача может потребовать написания сотен строк кода. Кроме того, абсолютно стандартные и однотипные операции с файлами требуют достаточно трудоемкой и кропотливой работы.

К сожалению, при выполнении подобных операций оказываются забытыми замечательные идеи, реализованные в операционной системе UNIX – составление маленьких программ «на лету», конвейерная обработка данных, использование специализированных программ работы с файлами.

В этой статье мы попытались применить эти концепции к созданию конвертора из формата MARC в разработанную нами автоматизированную картотеку книгообеспеченности на основе использования программы AWK.

Программа AWK была создана почти 30 лет назад и входила во все версии UNIX. Название программы – первые буквы фамилий авторов (Aho, Weinberg, Kernigan). Мы будем использовать GNU (Gnu's not UNIX) версию программы, которая распространяется бесплатно. Программа перенесена в операционную среду Windows. Она называется gawk.exe.

Статья ориентирована на программистов, знающих язык C или один из его клонов (JavaScript, PHP и т.д.).

Все тексты программ и исходные файлы данных можно найти на сайте журнала **www.lib-journal.ru**. Информация организована следующим образом – в главном каталоге находится файл gawk.exe и 4 подкаталога (part1, part2, part3, part4), в которых расположены исходные файлы и тексты программ. Для работы требуется только сама программа и любой текстовый редактор. Однако мы настоятельно рекомендуем использовать файл-менеджер FAR.

Начало работы

Итак, начнем. Во избежание обвинений в нарушении авторских прав разработчиков других программ воспользуемся свободно распространяемой библиотечной программой ISIS.

Из данной программы экспортируем файл из двух записей в формат MARC. Такой файл называется еще ISO-файлом. Исходный файл находится в каталоге part1 и называется src.iso. Сразу же выясняется, что ISIS изуродовал файл и поставил концы строк через каждые 80 символов (видимо, для удобства пользователя). Сделаем из этого обрезанного файла нормальный.

Для решения этой задачи нам потребуются еще два файла.

1. Командный файл запуска AWK – **go.bat.**

```
..\gawk -f del_cr.awk src.iso >cor.iso
```

..\gawk – имя запускаемой программы (**..** означает, что мы запускаем программу из внешнего каталога).

-f del_cr.awk - указывает AWK, что нужно выполнить программу, которая находится в файле del_cr.awk.

src.iso - имя входного файла.

>cor.iso – выводимую информации нужно направлять в файл cor.iso.

2. Программа для AWK – **del_cr.awk.**

```
{ printf("%s",$0); }
```

Эта программа считывает каждую строку из исходного файла и записывает в выходной файл всю информацию, за исключением символов конца строки. Все это делается одной функцией **printf** (работает так же, как и в C).

Анализ формата MARC

Теперь попробуем разобраться с форматом MARC. Файлы формата MARC не являются текстовыми в строгом смысле этого слова, однако возможностей AWK вполне достаточны и для работы с этими файлами.

Запись в формате MARC начинается с маркера (24 символа). Затем идет справочник, который состоит из нескольких статей, длиной 12 символов (по количеству полей в записи). Первые три символа в статье – номер поля. В конце справочника ставится символ - разделитель полей. Дальше записывается содержание полей, разделенных на подполя. После каждого поля ставится разделитель полей. В конце записи ставится символ – разделитель записей.

Рассмотрим общую структуру программы AWK.

```
BEGIN { действие }
шаблон { действие }
шаблон { действие }
      …
END { действие }
```

Перед началом обработки файла выполняется «действие» (последовательность операторов), указанное после ключевого слова BEGIN. Каждая строка из входного файла проверяется на соответствие «шаблону». Если строка соответствует, то для нее выполняется соответствующее «действие». Если шаблон не указан, то «действие» выполняется для всех строк.

После завершения обработки файла выполняется «действие», указанное после ключевого слова END.

Перед обработкой считанная строка делится на подполя (по умолчанию разделитель подполей - пробел). При этом создаются следующие переменные:

NF – число подполей,

$0 – вся строка,

$1 … $NF – подполя.

Можно изменить ограничитель строк и разделитель подполей. Для анализа ISO файла нам это потребуется.

В каталоге part2 находится файл **get_marc.awk**. Назначение этой AWK-программы – перевести информацию из формата MARC в более удобный для дальнейшей обработки вид.

Перед началом чтения файла в формате MARC определим разделители записей и подполей. ISIS выдает записи, разделенные символом '#' (мы присваиваем этот символ специальной переменной RS).

Для разделения полей используется символ '$' (мы присваиваем этот символ специальной переменной FS).

```
BEGIN{
  FS = "$";
  RS = "#";
}
```

Для каждой строки входного файла выполняются следующие действия (шаблон опущен).

```
{
```

Присваиваем переменной tmp значение первого подполя. Отметим, что в AWK есть переменные двух типов – числовые и строковые. Тип переменной может изменяться новым присваиванием. Описывать переменные заранее не нужно.

```
  tmp = $1;
```

Теперь нам нужно удалить маркер записи, который занимает первые 24 позиции. Для этого воспользуемся специальной функцией sub, которая заменяет символы в строке на другие. Функция принимает три параметра:

1. Описание информации, которую нужно заменить.

2. Строка символов, на которую надо заменить информацию, описанную в параметре 1.

3. Строка, в которой нужно произвести замену.

Параметр 1 – регулярное выражение (ограниченное знаками '/'). Использование регулярных выражений, может быть, самая привлекательная черта AWK. Дать полное описание регулярных выражений в рамках данной статьи невозможно, поэтому мы вынуждены ограничиться лишь объяснением того, что каждое выражение означает. В данном случае выражение **/^.{24}/** означает «любые 24 символа в начале строки». Эти любые 24 символа мы меняем на пустую строку в строке tmp.

```
  sub(/^.{24}/,"",tmp);
```

Прочитаем номера полей из справочника. Для этого запишем первые три символа в массив номеров полей, а затем удалим всю статью (12 символов). Воспользуемся стандартной логической конструкцией языка C while.

```
  i = 0;
  while(tmp != ""){
```

nfld – массив AWK. Предварительное описание массива не требуется.

i – текущий элемент массива.

substr(tmp,1,3) – функция, которая выделяет подстроку из строки tmp, начиная с 1-го символа и длиной 3 символа.

```
    nfld[i++] = substr(tmp,1,3);
```

Удаляем статью из справочника точно так же, как мы удаляли маркер записи.

```
    sub(/^.{12}/,"",tmp);
  }
```

Далее необходимо совместить номер поля и его содержание, которые создатели формата MARC заботливо записали в разных местах, наверняка, мотивируя это какими-то высшими соображениями.

При чтении записи формата MARC AWK записала в $1 маркер и справочник, а в $2 ..$NF – содержание полей. Воспользуемся стандартной конструкцией языка C for.

```
  for(i=2; i<=NF; i++)
```

Вывод информации мы осуществляем с помощью специального оператора AWK print. При выводе использована еще одна специфическая операция AWK – конкатенация (сложение) строк.

Эта операция обозначается пробелом. Кроме того мы не выводим пустые строки, если по какой-то причине они появятся.

```
    if($i != "") print nfld[i-2] $i;
```

В конце каждой записи выведем специальную строку. NR – это номер текущей записи (определяется AWK).

```
  print "***** End of record N " NR;
}
```

Запускаем программу с помощью файла **go.bat**.

Первая строка такая же, как и в предыдущем примере.

```
..\gawk -f del_cr.awk src.iso >cor.iso
```

Теперь мы из файла cor.iso сделаем файл cor.txt, который уже имеет удобочитаемый вид. Отметим использование опции

--re-interval

Эта опция необходима для правильной интерпретации регулярного выражения **(/^.{24}/.**

```
..\gawk --re-interval -f get_marc.awk cor.iso >cor.txt
```

В этом bat-файле реализован принцип конвейера, когда результаты работы одной программы сразу же обрабатываются другой (в нашем случае обе эти программы – AWK с разным набором выполняемых инструкций).

После выполнения go.bat создается файл cor.txt, в котором записано следующее.

```
011^aМальцев^cВ.А.
015^aОсновы политологии^eУчеб. для вузов
016^a2-е изд., доп^dМ.^fИТРК РСПП^h1997
017^a480 с.^cил
021^c5-88010-030-8^e12грн.40к.^g10 000
008^a32
009^a66.0я73
005^bПолитология - Учебники и учебные пособия
072^a1:Общественные нау-
ки^b1:книга^c1:русс.яз.^d2:Учебные издания
069^a66.0^bМ 21
078^aПолитология^bПолитика^cВласть политиче-
ская^dПолитическая система общества^eДемократия полити-
ческая^fМеждународные отношения^iКультура политиче-
```

```
ская^jКонфликты политические^gГлобальные проблемы обще-
ства^hУчебники
073^aч/з
074^aч/з^b427793^k1
080^b102
999^a01-23-02  11:12:51
***** End of record N 1
015^aПолитология^eКурс лекций^gМГУ
им.М.В.Ломоносова^iПод ред. М.Н.Марченко
016^a2-е изд., перераб. и доп^dМ.^fЗерцало^h1997
017^a383 с
021^c5-7218-0081-х^e11грн.20к.^g30 000
021^c5-88746-008-2
009^a66.0я73
005^bПолитология - Учебники и учебные пособия
072^a1:Общественные нау-
ки^b1:книга^c1:русс.яз.^d2:Учебные издания
069^a66.0^bП 50
078^aПолитология^bВласть политическая^cКультура полити-
ческая^dПсихология политическая^eПрава челове-
ка^fДемократия политическая^iПолитическая система обще-
ства^jЛичность^gКурс лекций
073^aч/з
074^aч/з^b427795^k1
080^b102
999^a01-23-02  11:28:31
***** End of record N 2
```

Потренируемся на простом примере.

Прежде чем создавать конвертор, решим более простую (но тоже полезную задачу). Выберем из файла все ключевые слова и рубрики. В каталоге part3 находится файл **get_kw.awk**, который выбирает из файла ISO все рубрики и ключевые слова. В нашем примере ключевые слова – все подполя поля 078, а рубрики – все подполя поля 005.

ISIS использует в качестве разделителя подполей символ ‘^’. Присвоим его переменной FS.

```
BEGIN{
  FS = "^";
}
```


Для хранения ключевых слов и рубрик мы будем использовать ассоциативные массивы. В AWK индексом массива может быть не только целое число, но и любая строка. Обращение к элементу такого массива происходит точно также, как и к элементу обычного массива. Ключевые слова мы будем хранить как индексы массива **kw**, а рубрики – как индексы массива **rb**.

Всю работу по анализу того, было данная рубрика уже записана, или нет, берет на себя AWK. Нам нужно только что-нибудь присвоить конкретному элементу массива **rb**. Так как все равно нужно что-нибудь присваивать, мы можем попутно определить число повторений каждой рубрики (ключевого слова).

Начинаем обработку строк.$1 – номер поля. Проверяем, не равен ли номер поля 5. Обратите внимание, что AWK автоматически выполняет нужное преобразование типов.

```
{
```

Обрабатываем рубрики.

```
  if($1 == 5)
    for(i=2; i<=NF; i++){
      s = $i;
```

Первый символ в $2 … $NR – код подполя. Удаляем его (любой символ в начале строки).

```
      sub(/^./,"",s);
      if(s != "") rb[s]++;
    }
```

Аналогично обрабатываем ключевые слова.

```
  if($1 == 78)
    for(i=2; i<=NF; i++){
      s = $i;
      sub(/^./,"",s);
      if(s != "") kw[s]++;
    }
}
```

После обработки всех записей файла выводим ключевые слова и рубрики в два разных файла. Конструкция **for(i in kw)** просто перебирает в цикле все элементы массива. Вывод информации перенаправляется в файл **kw.txt** для массива ключевых слов

и в файл **rb.txt** для массива рубрик. Повторяемость и ключевое слово (рубрика) разделены табуляциями. Эти файлы можно экспортировать в Excel.

```
END{
for(i in kw) printf("%d\t%s\n",kw[i],i) >"kw.txt";
for(i in rb) printf("%d\t%s\n",rb[i],i) >"rb.txt";
}
```

Запускаем программу с помощью файла **go.bat**.

```
..\gawk -f del_cr.awk src.iso >cor.iso
..\gawk --re-interval -f get_marc.awk cor.iso >cor.txt
..\gawk -f get_kw.awk cor.txt
```

В результате получаем файл **rb.txt**

```
2       Политология - Учебники и учебные пособия
```

и файл **kw.txt.**

```
1       Психология политическая
2       Политическая система общества
2       Демократия политическая
2       Власть политическая
1       Права человека
1       Курс лекций
1       Конфликты политические
2       Культура политическая
1       Международные отношения
1       Глобальные проблемы общества
1       Учебники
1       Личность
1       Политика
2       Политология
```

Конвертор

Для начала четко поставим задачу. Нам нужно вытащить информацию из ISO-файла и представить ее в виде двенадцати строк. Подробнее об экспорте информации в автоматизированную картотеку книгообеспеченности см. в инструкции:

www.lib-journal.ru/exlibris/instr.html

С помощью таблицы определения полей ISIS поставим в соответствие каждой строке выходного файла номера полей и подполей входного файла ISO.

№	Необходимая информация	Представление в формате MARC
1	Авторы	11 - Индивидуальный автор a - Фамилия автора c - Инициалы
2	Название	15 - Область заглавия a - Основное заглавие e - Сведения, относящиеся к заглавию g - Первые сведения об ответственности i - Вторые и последующие сведения об ответственности
3	Место издания	16 - Область издания и выходные данные d - Первое место издания
4	Издательство	16 – Область издания и выходные данные f - Изд-во или издающая организация
5	Год издания	16 – Область издания и выходные данные h - Дата издания
6	Количество страниц	17 - Количественные характеристики a - Объем
7	УДК/ББК – код	9 - Индекс ББК a - Основной индекс ББК
8	УДК/ББК - авторский знак	69 - Шифр хранения b - Авторский знак
9	Адрес книги	Не используется
10	Количество экземпляров	74 - Учет инвент. док-тов k - Общее количество экземпляров
11	Тип издания	Не используется
12	Строка "Конец записи"	

Конвертор находится в файле **export.awk** в каталоге part4.

Программа начинается с нового понятия – функции, определенной пользователем. Описание функции должно находиться в начале файла. Описания функций практически не отличаются от описаний функций в C. Единственное принципиальное отличие – **все переменные имеют глобальную область видимости**. Этим обусловлены «странные» названия внутренних переменных (например, **get_subfield_sym**).

Функция **get_subfield** возвращает значение подполя (символ подполя задается параметром) в текущей записи или пустую строку, если подполе в текущей записи отсутствует.

```
function get_subfield(sym){
  for(i=2; i<=NF; i++){
    get_subfield_sym = substr($i,1,1);
    if(get_subfield_sym == sym){
      get_subfield_tmp = $i;
      sub(/^./,"",get_subfield_tmp);
      return get_subfield_tmp;
    }
  }
  return "";
}
```

Программа начинается стандартно (см. предыдущие примеры).

```
BEGIN{
  FS = "^";
}
```

В конверторе мы будем использовать шаблоны. Начнем с заполнения поля «Автор». Регулярное выражение в шаблоне обозначает все строки, которые начинаются с **011**. Поле может повторяться. В этом случае мы разделяем авторов запятой.

Следует обратить внимание на оператор **next**. Он производит переход к следующей записи, при этом все оставшиеся шаблоны не обрабатываются. Мы применяем этот оператор в том случае, если подполе ‘a’ (фамилия автора) не найдено.

```
/^011/{
  tmp = get_subfield("a");
  if(tmp == "") next;
  tmp = tmp " " get_subfield("c");
  if(s01_author != "") s01_author = s01_author ",";
```



```
  s01_author = s01_author tmp;
}
```

Произведем обработку информации по названию книги.

```
/^015/{
  tmp = get_subfield("a");
  if(tmp == "") next;
  tmp1 = get_subfield("e");
  if(tmp1 != "") tmp = tmp ". " tmp1;
  tmp1 = get_subfield("g");
  if(tmp1 != "") tmp = tmp ". " tmp1;
  tmp1 = get_subfield("i");
  if(tmp1 != "") tmp = tmp "/ " tmp1;
  s02_name = tmp;
}
```

Из поля 016 мы выбираем информацию о месте издания, издательстве и дате издания. Обратите внимание на функцию **gsub**. Она отличается от функции **sub** тем, что меняются все найденные подстроки, описанные параметром 1. Регулярное выражение в параметре 1 обозначает «все символы, которые не являются цифрами». Смысл этой замены понятен – в поле может быть введено, например, «2001 г.», а автоматизированная картотека книгообеспеченности ожидает в этой строке число.

```
/^016/{
  s03_place = get_subfield("d");
  s04_publ  = get_subfield("f");
  s05_year  = get_subfield("h");
  gsub(/[^0-9]/,"",s05_year);

}
```

Запоминаем количество страниц.

```
/^017/{
  s06_pages = get_subfield("a");
  gsub(/[^0-9]/,"",s06_pages);
}
```

УДК/ББК - код

```
/^009/{
  s07_code = get_subfield("a");
}
```

УДК/ББК – авторский знак.

```
/^069/{
  s08_as = get_subfield("b");
}
```

Количество экземпляров.

```
/^074/{
  s10_quan = get_subfield("k");
  gsub(/[^0-9]/,"",s10_quan);
}
```

Конец записи – строка, начинающаяся с пяти звездочек (мы ее сами так определили в программе get_marc). Регулярное выражение в шаблоне обозначает «пять звездочек в начале строки». Обратите внимание, что каждая звездочка «защищена от интерпретации (экранирована)» символом '\'. Звездочка в регулярных выражениях имеет специальный смысл. Поэтому мы должны дать понять AWK, что имеем в виду именно звездочку, а не «любое количество символов».

Обработка конца записи во входном файле проста – мы просто выводим ранее сформированные значения в выходной файл. Затем «обнуляем» соответствующие переменные для обработки новой записи. Символом '#' начинается комментарий.

```
/^\*\*\*\*\*/{
  print s01_author; # Авторы
  print s02_name;   # Название
  print s03_place;  # Место издания
  print s04_publ;   # Издательство
  print s05_year;   # Год издания
  print s06_pages;  # Количество страниц
  print s07_code;   # УДК/ББК - код
  print s08_as;     # УДК/ББК - авторский знак
  print s09_addr;   # Адрес книги
  print s10_quan;   # Количество экземпляров
                    # Тип издания
  if(s11_pbtp == "") s11_pbtp = -1;
  print s11_pbtp;
  print $0;         # Конец записи
  s01_author = "";
  s02_name   = "";
  s03_place  = "";
  s04_publ   = "";
  s05_year   = "";
  s06_pages  = "";
  s07_code   = "";
  s08_as     = "";
  s09_addr   = "";
  s10_quan   = "";
  s11_pbtp   = "";
}
```


Запускаем программу с помощью файла **go.bat**.

```
..\gawk -f del_cr.awk src.iso >cor.iso
..\gawk --re-interval -f get_marc.awk cor.iso >cor.txt
..\gawk -f export.awk cor.txt > exl.txt
```

В результате получаем файл **exl.txt**.

```
Мальцев В.А.
Основы политологии. Учеб. для вузов
М.
ИТРК РСПП
1997
480
66.0я73
М 21

1
-1
***** End of record N 1

Политология. Курс лекций. МГУ им.М.В.Ломоносова/ Под
ред. М.Н.Марченко
М.
Зерцало
1997
383
66.0я73
П 50

1
-1
***** End of record N 2
```

Перед экспортом в автоматизированную картотеку книгообеспеченности нам нужно разобраться с русскими кодировками. ISIS работает с кодировкой DOS (как и любая программа под DOS), а картотека (как программа под Windows) требует информацию в кодировке Windows-1251.

Эту проблему мы решаем с помощью программы FAR. Алгоритм действий следующий:

1. скопировать все строки из файла exl.txt в буфер обмена;
2. создать новый файл;
3. в новом файле переключить кодировку на «Win»;
4. вставить информацию из буфера обмена.

Заключение

Кроме текстов программ и исходных файлов на сайте журнала помещены следующие материалы:

- полное и очень хорошее описание AWK на английском языке;
- фрагмент руководства пользователя Linux, содержащий описание AWK на русском языке.

Исходя из нашего опыта, настоятельно рекомендуем ознакомиться с описаниями и изучить программу AWK. Это позволит в дальнейшем сэкономить массу времени. Кроме того, сфера применения программы не ограничивается библиотечным делом. Использование AWK при создании web-сайтов может позволить отказаться от применения баз данных на стороне сервера.

КНИЖНЫЙ РЫНОК И БИБЛИОТЕКИ

У НАС В ГОСТЯХ

Светлана Зорина – руководитель отдела рекламы и PR издательского торгового дома «КноРус», главный редактор газеты «КноРус»

Светлана Юрьевна, прежде всего познакомьте наших читателей со структурой и задачами вашей организации.

Издательский торговый дом «КноРус» создан в 1993 году. Он представляет собой холдинг, в состав которого на правах самостоятельных юридических лиц входят изда-