Применение OCR в технологии каталогизации на примере разработки специализированного программного модуля для ИРБИС64

Using OCR in cataloguing technology as exemplified in the development of specialized IRBIS64 software module

Застосування OCR у технології каталогізації на прикладі розробки спеціалізованого програмного модуля ІРБІС64

А. В. Шувалов
Саратов, Россия

Arseny Shuvalov,
Saratov, Russia

Арсеній В. Шувалов
Саратов, Росія

В настоящее время практически во всех библиотеках используются информационные системы, служащие для автоматизации библиотечных процессов. Одними из самых рутинных, слабо автоматизированных до сих пор являются процессы по непосредственному формированию содержания электронного каталога. Они подразумевают последовательное заполнение отдельных полей записи, таких как имя автора, название книги, сведения об издании и др. Предлагаемый доклад содержит обзор средств, призванных облегчить процесс каталогизации и передать ЭВМ часть функций по заполнению полей при составлении записей. Применяемый автором подход основан на использовании технологии OCR и собственных программных решениях.

Today, practically every library uses information system to computerize its operations. Acquiring the content of electronic catalogs is still the most routine, poorly computerized processes. It implies sequential filling in individual record fields, such as author's name, title, edition details, etc. The author reviews the instruments to facilitate cataloguing by computerizing field filling-in. The suggested approach is based on OCR technology and independent author software solutions.

Сьогодні практично у всіх бібліотеках використовуються інформаційні системи для автоматизації бібліотечних процесів. Одними із самих рутинних, слабко автоматизованих до теперішнього часу є процеси з безпосереднього формування змісту електронного каталогу. Вони передбачають послідовне заповнення полів запису, таких як ім'я автора, назву книги, відомостей про видання та ін.. Пропонована доповідь містить огляд засобів, що повинні полегшити процес каталогізації та передати ЕОМ частину функцій по заповненню полів при складанні записів. Підхід, що застосовує автор, базується на використанні технології OCR та власних програмних рішеннях.

Пожалуй, до сих пор одной из наиболее трудоёмких задач в процессе автоматизации библиотек является создание и заполнение содержимым библиографических записей, т.е. непосредственно каталогизация. Как мы покажем, скорость работы специалиста-библиографа, осуществляющего каталогизацию книжных единиц, может существенно повыситься без заметного ухудшения качества за счёт оснащения библиотек сканерами и специализированным ПО.

Как известно, каталогизация подразумевает последовательное заполнение отдельных полей библиографической записи, таких как имя автора, название книги, сведения об издании, количественные характеристики. Для журналов и сборников научных трудов, а также для других изданий, содержание которых предполагает деление на отдельные смысловые единицы, различающиеся по тематике и часто имеющие различных авторов, необходимо также частично или полностью делать роспись оглавления. До настоящего момента эта работа во многом представляет собой рутинный, слабо автоматизированный процесс. Безусловно, технология заимствования записей во многом способствовала облегчению труда библиографа, но вместе с тем, надо отдавать себе отчёт в том, что ещё не все библиотеки могут позволить себе участие в ИРБИС-корпорации и MARC. К тому же, сам принцип работы корпорации предполагает, что вы должны не только брать, но и отдавать что-то взамен, следовательно, какое-то количество записей вам всё равно придётся создавать

самостоятельно. Чем быстрее они будут созданы, тем скорее другие участники корпорации смогут воспользоваться результатом этого труда. А значит, применение технологий сканирования и распознавания текста является уместным и в рамках участия в ИРБИС-корпорации.

Мы будем рассматривать процесс каталогизации на примере АБИС «ИРБИС64» и его модуля «Каталогизатор». Для начала проанализируем те средства автоматизации, которые нам доступны уже сейчас.

1. Раскрывающиеся списки, позволяющие заполнять необходимые поля на основе уже имеющихся в базе записей, содержащих такие же или схожие данные для этих полей (например, то же издательство – тогда его название не нужно вводить целиком, а достаточно выбрать из списка после ввода первых букв названия).

Итог: Облегчают работу, но все новые данные приходится вводить вручную.

2. Применение буферной записи для создания новых библиографических записей на основе уже имеющихся. Безусловно, чем больше общих данных содержат записи, тем быстрее будет осуществляться ввод. Данный подход имеет свои подводные камни, служащие потенциальным источником ошибок, выражающиеся в том, что старая запись копируется целиком со всеми её полями, в том числе с теми, значение которых необходимо исправить. Здесь нужно учитывать человеческий фактор, вследствие которого оператор может просто забыть изменить значения некоторых нужных полей.

Итог: Не автоматизирует ввод новых данных и способствует возникновению ошибок.

3. Импорт готовых записей из других баз данных. Это могут быть как базы, созданные в самом ИРБИСе, так и базы данных других АБИС или произвольных СУБД. Безусловно, данная технология эффективнее, чем создание новых записей «с нуля», и её стоит применять во всех случаях, когда требуемые записи уже имеются в каком-либо формате. Но зачастую она требует написания специализированных конверторов и, главное, – не решает проблему первоначального источника получения записей. Другими словами, человек, который прежде занимался созданием данных записей, сам должен был решить для себя этот вопрос. Сюда же можно отнести и случай импорта готовых записей вследствие участия в корпорации.

Итог: Не решает проблему первоначального источника получения записей, зачастую требует создания специализированного ПО для данного конкретного случая.

4. Копирование содержимого отдельных полей записи через буфер обмена Windows. При этом пользователю самому приходится выбирать те ячейки таблицы рабочего листа записи в Каталогизаторе, куда необходимо вставить текст. К тому же, текстовые данные, вставляемые из буфера обмена, должны быть откуда-то уже перенесены. Если не рассматривать случай переноса библиографических записей по отдельным частям из какой-то другой базы, как в 3-м варианте (это оправдано лишь в случае небольшого количества таких записей, иначе целесообразнее применять конверторы) или, скажем, с каких-нибудь страниц Интернета, то нам так или иначе придётся задействовать сканер или цифровой фотоаппарат и программы OCR (оптического распознавания символов) – чтобы получать эти данные непосредственно со страниц издания, а для этого, как мы покажем, эффективнее применять программные средства вроде тех, что описываются в данной работе.

Итог: Требует применения сканера и программных средств распознавания текста, но не лишает оператора необходимости непосредственного участия в процессе ввода записи, поскольку именно ему приходится каждый раз выбирать нужные поля таблицы для заполнения их данными.

Забегая вперёд, отметим, что наш собственный подход является в какой-то мере совмещением 3-го и 4-го вариантов. Текстовые данные для формируемой записи, получаемые со страниц книжных изданий посредством сканера и OCR, обрабатываются так же, как если бы они брались из какой-либо базы данных, а используемые программные средства являются по сути конверторами

из формата представления этих данных в книжных изданиях в формат ИРБИСа. В случае обработки сводного библиографического описания, обычно располагающегося на обороте титульного листа, такой конвертор может работать и без участия человека; в случае обработки оглавления его предварительно необходимо настраивать.

Итак, можно сделать вывод, что имеющиеся в модуле Каталогизатор ИРБИС 64 средства не обеспечивают автоматизацию самого процесса формирования библиографической записи. Впрочем, насколько известно автору, и в других автоматизированных библиографических системах эта задача оставляется на усмотрение сторонних разработчиков. Некоторые подобные средства, демонстрирующие применение сканера и технологий OCR, в частности, при решении задачи ретроконверсии карточных каталогов, действительно были созданы, но, судя по всему, разработанные программные средства так и не стали достоянием широкой общественности. Примерами могут служить разработки отечественных фирм «ГИПЕР» (http://www.gpntb.ru/win/inter-events/crimea2004/300.pdf), «ПроСофт-М» (https://eva.rsl.ru/old/2000/eva/200008/lavrionova-r.htm) и белорусской «Агат-Систем» (http://www.agat-system.com/direction/it/itsr/). Отметим, что независимо от того, как обстоят дела с распознаванием каталожных карточек, которые могут иметь большое количество вариантов оформления, сводное библиографическое описание, которым снабжается практически каждое современное издание, достаточно стандартизовано, и если не браться за разработку некого всеобъемлющего безошибочного алгоритма на все случаи жизни, то выделение в нем отдельных полей представляет собой достаточно тривиальную задачу. Причиной, по которой подобная технология до сих пор не встроена непосредственно в АРМ «Каталогизатор», может являться повышение стоимости конечного продукта вследствие включения в него лицензии на систему OCR, а также слишком узкий сектор потенциальных пользователей: известно, что в настоящее время не во многих библиотеках нашей страны вообще используются сканеры. Но поскольку многие сканеры уже имеют в комплекте программу распознавания символов (чаще всего это ABBYY FineReader), то решение второй проблемы повлечёт за собой и автоматическое решение первой – ведь текстовые данные для библиографической обработки (представления в виде отдельных полей библиографической записи) необязательно должны формироваться самой программой путём внутреннего вызова функций OCR, они могут переноситься посредством буфера обмена и из самого приложения FineReader или подобного ему. Дело лишь за тем, чтобы оснастить библиотеки сканерами с входящими в комплект программами для OCR. Сама библиографическая обработка распознанного текста может осуществляться небольшими программными модулями, подключаемыми в качестве плагинов к различным автоматизированным рабочим местам (АРМ) конкретной АБИС, таким как АРМ Каталогизатор для ИРБИС 64, либо приложениями, оформленными в виде самостоятельных АРМ. Одним из примеров последних является АРМ «АльтерВвод» (от слов «альтернативный ввод»), специально разработанный для ИРБИС 64 автором данной работы.

АРМ «АльтерВвод» представляет собой упрощённый клиент для каталогизации, заточенный на импорт записей из внешних источников. Для этого он обладает как возможностями подключения плагинов в привычном формате ИРБИС 64, так и собственным встроенным библиографическим редактором. Для него было разработано внешнее приложение БАРТ (Библиографический Анализ Распознанного Текста) в формате обычного плагина ИРБИС 64 в расчёте на то, что его можно будет вызывать и из АРМ Каталогизатор. Оно позволяет импортировать практически готовую запись из предварительно отсканированных и распознанных текстовых данных непосредственно в АБИС. Данное приложение основано на том, что сводное библиографическое описание, помещаемое на обороте титульного листа или в конце всех современных изданий, имеет достаточно чёткую структуру и без особых проблем автоматически разбивается на отдельные поля, из совокупности которых можно сразу же составить требуемую запись. Следовательно, необходимо было только реализовать соответствующий алгоритм. Отмечу, что мой собственный алгоритм, конечно, не идеален и не учитывает всех возможных вариантов структурирования элементов, но, безусловно, способствует ускорению работы по каталогизации – от оператора требуется только поправить некоторые возможные ошибки, возникшие в ходе восстановления структуры записи.

В дополнение к возможности подключения внешних плагинов в формате ИРБИС64, АРМ «АльтерВвод» предлагает и свой собственный текстовый редактор для выделения в тексте требуемых элементов библиографической записи с последующим переносом их в соответствующие поля текущей записи – так называемый *библиографический редактор*. Кроме простого переноса текстовых данных из программы OCR посредством буфера обмена Windows реализована также возможность подключения внешних плагинов к библиографическому редактору, на выходе которых в программу передаются готовые текстовые данные. Таким образом, пользователь может создать свой собственный встраиваемый программный инструмент, реализующий OCR на базе любой из существующих для этой цели библиотек (*Одно из таких решений будет продемонстрировано мной на конференции*), и ценовая политика конечного продукта АРМ «АльтерВвод», таким образом, не будет подразумевать жёсткой привязки к созданным разработчиками интегрированным средствам OCR (т.е. библиотекам не придётся платить за лицензирование встроенного движка, если они, к примеру, уже приобрели сканер или сканирующую ручку с входящим в его комплект приложением FineReader).

Для восстановления структуры библиографической записи из текстовых данных, поступающих на вход библиографического редактора, был создан специальный язык BIRMA (Bibliographic Information Recognition MArkup Language), или БИРМА (Библиографический Инструмент Распознавания Маркировки). Соответствующая программная библиотека BIRMA.DLL содержит ряд функций для библиографического распознавания полей записи и может подключаться к произвольным разрабатываемым программам, требующим поддержки этого языка. Язык BIRMA по своей структуре похож на язык форматирования ISIS, хотя выполняет, можно сказать, противоположную задачу: не формирует текст из элементов библиографической записи, а, наоборот, служит для составления запросов, осуществляющих выделение этих отдельных элементов из исходного текста. Правила этого языка лучше всего рассмотреть на примере конкретной задачи, а именно – восстановления структуры оглавления. Предположим, имеем такой текст оглавления (данный текст был взят из сборника научных статей):

Восстановить структуру данного оглавления и перевести его в формат ИРБИС нам поможет следующий запрос на языке BIRMA:

```
(&call_func('partmarker', '<буква>.{ }<буква>.')+(v330^F)&call_func('partmarker',
'<Ввод>'), &call_func('partmarker', '{ }')v330^C'\t', &call_func('partmarker',
'<цифра>')+v330^4&call_func('partmarker', '<Ввод>'))
```

Как видим, запрос состоит из нескольких предложений – подзапросов, разделённых запятой и обрамлённых круглыми скобками. Внешние скобки означают, что все содержащиеся в них отдельные запросы должны выполняться циклически, пока не достигнут конец текста. Каждый запрос может включать в себя идентификатор искомого поля записи, за которым может следовать разделитель его конкретного подполя. В нашем случае в каждом из запросов, содержащихся в скобках, имеется подобный элемент: v330^F, v330^C и v330^4. Это означает, что мы ищем в тексте элементы, соответствующие трём подполям поля оглавления 330: ФИО первого автора, заглавию и номеру страницы. Поскольку эта группа запросов заключена в круглые скобки, они выполняются последовательно до конца исходного текста с увеличением счётчика повторений поля 330 на каждом шаге цикла. Таким образом, мы последовательно просматриваем весь текст и выделяем из него нужные элементы для каждого из пунктов оглавления. Мы видим, что первая конструкция **(v330^F)** также заключена в скобки. Это означает, что данный элемент является необязательным, т.е. на каждом из повторений поле 330 может либо содержать, либо не содержать подполе F (ФИО). Если в процессе анализа текста следующий обязательный элемент (который запишется в v330^C) найден раньше, чем очередной необязательный элемент (v330^F), считается, что в данном повторении поля его нет. В нашем случае мы видим, что текст оглавления начинается с заголовка раздела, т.е. на данном повторении мы сразу же выделяем заглавие и не вводим никаких данных в подполе ФИО.

Справа и слева от идентификаторов конкретного поля/подполя должны стоять текстовые фрагменты, содержащие последовательности символов, составляющих границы данного элемента в исходном тексте. Другими словами, справа и слева от выделяемых элементов в исходном тексте должны содержаться эти последовательности. В качестве составляющих их символов могут выступать, например, символы пробела, табуляции, точки и др. Возможно использование регулярных выражений, в том числе в виде выражений на специально разработанном автором языке ПАРТИЗАН (Парсер-Анализатор Регулярной Текстовой Информации с Заложенной Автоматизацией Набора). Встречающиеся слева и справа от идентификаторов поля выражения вида &**call_func**('partmarker', **expr**) говорят о том, что вызывается функция partmarker, аргумент которой **expr** как раз представляет собой выражение языка ПАРТИЗАН. В первом подзапросе выражение **<буква>.{ }<буква>.** обозначает две подряд идущие буквы русского языка с точками после каждой, между которыми может находиться любое количество пробелов (или ни одного). Стоящее справа в этом запросе выражение **<Ввод>** означает просто перевод строки (при анализе заменяется на последовательность управляющих символов **\r\n**). Таким образом, весь запрос можно расшифровать так: мы считаем значением для ФИО в каждом из повторений поля текстовый фрагмент, заключённый между инициалами (**<буква>.{ }<буква>.**) и символами перевода строки. Знак «+», помещённый после выражения, обозначающего левую границу, и перед идентификатором поля, означает, что последовательность символов, которой соответствует данное выражение (**<буква>.{ }<буква>.**) на текущем повторении цикла в исходном тексте, включается в формируемый текстовый фрагмент (т.е. в данном случае инициалы служат левой границей для выделения значения ФИО, но с них же и должно начинаться формируемое значение). Знак «+», стоящий после идентификатора поля/подполя, будет означать точно такую же необходимость включения идущего следом за ним в запросе значения правой границы в формируемое значение соответствующего текстового фрагмента.

Второй подзапрос содержит такие выражения для границ соответствующего найденного фрагмента: **&call_func('partmarker', '{ }')** и **'\t'**. Это означает, что мы берём текстовый фрагмент, располагающийся после произвольного количества подряд идущих пробелов (подразумевается, что они стоят в начале строки) и до управляющего символа табуляции, и считаем его очередным

значением подполя C (заглавия) на данном повторении его поля 330. Сразу за ним следует запрос **&call_func('partmarker', '<цифра>')+v330^4&call_func('partmarker', '<Ввод>')**, означающий, что мы далее просматриваем исходный текст до появления в нём первой цифры, отсчитываем от неё (включая и саму эту цифру) текстовый фрагмент до появления управляющего символа перевода строки, и считаем его очередным значением подполя 4 (номер страницы) на данном повторении поля 330. Далее вся последовательность запросов повторяется для формирования следующего пункта оглавления.

Отметим, что в большинстве видов текста оглавления, помещаемого в начале или в конце издания, заглавие очередного пункта оглавления обычно отделяется от следующего за ним номера страницы отточием, т.е. множеством точек, заполняющих строку до её правого края, по которому располагается числовое значение номера страницы. Такой последовательности символов примерно соответствует выражение языка ПАРТИЗАН **.{.}** (или, чтобы не путать с точками внутри самого заглавия, – **..{.}**), но, как правило, существующие программы и программные библиотеки OCR заменяют эту последовательность на один или несколько символов табуляции. Так что запрос **&call_func('partmarker', '{ }')v330^C'\t'** подходит для множества случаев выделения заглавия / названия для статей или отдельных разделов сборников. Безусловно, для текстов оглавлений современных изданий характерны разнообразные варианты разметки, но практически все их случаи можно строго формализовать средствами языка BIRMA. Для сложных случаев предусмотрена команда ветвления, которая позволяет модифицировать в запросе значения границ для элемента на основании найденных значений границ предыдущих элементов или менять порядок следования элементов более сложным образом, чем это позволяют сделать круглые скобки, в которые заключён идентификатор поля. Однако, как показывает личная практика автора, даже применение одних только описанных простейших средств уже позволяет значительно автоматизировать весь процесс ввода практически любого оглавления. Время, требуемое на последующую обработку с исправлением ошибок и дополнительным вводом информации (некоторые поля вследствие неполноты применённого запроса могут оказаться не заполненными или заполненными частично), не идёт ни в какое сравнение с последовательным копированием и вставкой текстовых фрагментов в нужные подполя рабочего листа вручную. Про перепечатывание вручную всех пунктов оглавления посредством клавиатуры и говорить не приходится.

Подчеркну ещё раз, что при использовании традиционного подхода процесс росписи оглавления является еще более трудоемким, чем создание самой записи. Оглавление может состоять из нескольких страниц текста. Работа по обработке даже распознанного текста, но без применения специализированного ПО представляет собой чисто механический труд с многократным переносом нужных данных из буфера обмена в соответствующие поля таблицы рабочего листа. Таким образом, использование АРМ «АльтерВвод» и языка BIRMA является единственной на сегодняшний момент технологией быстрого ввода оглавления, хотя, безусловно, одним только оглавлением область их применения не исчерпывается.