abaraw: A shorthand notation for bird records

Zoological
Data Processing

Toward more efficient data entry

John W. Shipman

2013-02-07 13:30

Abstract

Describes a system for the rapid and efficient encoding of ornithological field records.

This publication is available in Web form[1] and also as a PDF document[2]. Please forward any comments to **john@nmt.edu**.

Table of Contents

1. Introduction

In the document *A system for encoding bird field notes*[3], we describe a technique for using an XML document type to represent field records of wild birds. The author has used this system for some time to publish his field notes[4].

Initially, the author used the `emacs` text editor and the related `nxml-emacs` package to enter and maintain the XML notes files; see the documentation for `nxml-emacs`[5].

However, for most notes, that system proved to be somewhat slow and cumbersome. The current document describes a "shorthanding" system that allows rapid creation of most of the XML from a terse textual notation that greatly speeds up the entry of *most* of the information.

The shorthand notation is not intended to create every type of information in the XML field notes schema. This schema provides for a wide variety of types of information: notes on breeding status, vocalization, and many other kinds of specialized content. However, the great bulk of the records are pretty simple—what kind of bird, how many.

[1] http://www.nmt.edu/~shipman/aba/raw/doc/
[2] http://www.nmt.edu/~shipman/aba/raw/doc/abaraw.pdf
[3] http://www.nmt.edu/~shipman/aba/doc/
[4] http://www.nmt.edu/~shipman/aba/field.html
[5] http://www.nmt.edu/tcc/help/pubs/nxml/

The author decided that a mixed strategy might be best: develop a terse shorthand notation that represents most of the data, and use that to create the bulk of the XML notes file; then use the full XML editor to add in anything that isn't captured in the shorthand notation.

Hence, the workflow goes like this:

1. Transcribe the field notes into the shorthand notation as a file whose name ends with the extension "`.in`". Omit the kinds of information that cannot be represented in the shorthand notation (breeding, vocalization, etc.).
2. Use the *abaraw* script to process that file, creating an XML file whose name ends with "`.out`".
3. Use `emacs` with `nxml-mode` to add any information that was omitted in the shorthand notation.

At this writing, the author has been using the shorthand system for several months, and finds that it greatly reduces the time required for data entry of field notes.

2. Installation

The directory in which this script is run must have the following files available, as either copies or soft links:

- `abaraw`[6]: The *abaraw* script itself.
- `abbr.py`[7]: Python module to process bird abbreviations.
- `aou.xml`[8]: XML file defining the latest AOU Check-List taxonomy.
- `rnc.py`[9]: Python constants for names from the schema.
- `rnc_txny.py`[10]: Used by `txny.py`.
- `txny.py`[11]: Module to read the taxonomy authority file.

Additionally, the Python `lxml` package must be installed; see *Python XML processing with lxml*[12].

3. Operation of the *abaraw* script

An input file must have a name of this form:

```
yyyy-mm.in
```

The `yyyy-mm` part gives the year and month of the notes. For example, a notes file for July 2008 must be named "`2008-07.in`".

To process one file, use this command:

```
abaraw yyyy-mm.in
```

Output is sent to the standard output stream. Typically you will want to redirect it to a file named *yyyy-mm*`.out`. For example, to process the July 2008 file:

[6] http://www.nmt.edu/~shipman/aba/raw/doc/ims/abaraw
[7] http://www.nmt.edu/~shipman/xnomo/ims2/abbr.py
[8] http://www.nmt.edu/~shipman/xnomo/aou/aou.xml
[9] http://www.nmt.edu/~shipman/aba/doc/pyims/rnc.py
[10] http://www.nmt.edu/~shipman/xnomo/ims2/rnc_txny.py
[11] http://www.nmt.edu/~shipman/xnomo/ims2/txny.py
[12] http://www.nmt.edu/tcc/help/pubs/pylxml/

```
abaraw 2008-07.in >2008-07.out
```

4. The input syntax

Each input file is a mixture of these kinds of lines:

- A *day line* starts with a bang ("`!`") character and defines the date, state code, and default locality for the lines that follow it, up to the next day line or to the end of the file. See Section 4.1, "Syntax of the day line" (p. 3).
- A *locality definition line* starts with an at-sign ("`@`") character and defines a new locality code that applies to following lines up to the next locality line, day line, or end of file. See Section 4.2, "The locality definition line" (p. 4).
- If the observer leaves a particular locality but then returns to that locality later in the same day, a *locality back reference* line specifies the locality that applies to following lines up to the next locality line, day line, or to the end of the file, whichever comes first. See Section 4.3, "The locality back-reference line" (p. 4).
- A *census line* describes the sighting of one or more kinds of birds. See Section 4.4, "The census line" (p. 4).

The reader may wish at this time to review the schema[13], since the output of the shorthanding system is destined for an XML file conforming to that schema. We will use names from the schema to indicate where shorthand items will go in the output.

4.1. Syntax of the day line

Here is the general form of a day line:

```
!state yyyy-mm-dd locality-def
```

`state`
: The state or region code, to be copied to the `state` attribute of `day-notes`.

`yyyy-mm-dd`
: The date of the following records; this will be copied to the `date` attribute of the `day-notes` element.

`locality-def`
: The rest of the line has the same format as a locality definition line, including the initial "`@`". This locality's code will be copied to the `day-loc` attribute of the `day-notes` element, and its definition will become a `loc` element. See Section 4.2, "The locality definition line" (p. 4).

Here are some examples of day lines.

```
!nm 2008-06-15 @SocCo Socorro County
!nm 2008-07-22 @BdA Bosque del Apache NWR
!ca 2009-07-10 @PAB Palo Alto Baylands
```

[13] http://www.nmt.edu/~shipman/aba/doc/schema.html

4.2. The locality definition line

To define the location of one or more sightings, a locality definition line defines a new locality code and the corresponding full description. This locality is applied to all the census lines that follow, up to the next locality line or day line (or to the end of the file).

Here is the format of this form of locality line:

```
@code loc-name
```

In the output file, the *code* is copied to the `code` attribute of the `loc` element, and the `loc-name` becomes the `name` attribute of that `loc` element.

Examples:

```
@WCRd Magdalenas: Water Canyon Road
@ABQ Albuquerque
@TerryL Socorro: New Mexico Tech: Terry Lake
```

4.3. The locality back-reference line

If the observer leaves a certain locality and then returns to it later, it is not necessary to repeat the entire locality definition line. Just use a locality back-reference line, which has this general form:

```
@code
```

The supplied locality *code* is applied to all following census lines up to the next locality line, day line, or end of file. The code must have been defined on a locality definition line used earlier in the same day.

Examples:

```
@SocCo
@BdA
@PAB
```

The back-reference form may not be used in a day line.

4.4. The census line

Any line that does not start with "`!`" or "`@`" is a census line. A census line consists of one or more *bird groups* separated by spaces.

Each bird group describes the sighting of one particular kind of bird. Each group results in the output of one `form` element in the XML output, and has this syntax:

- A *bird ID group* that describes the taxonomic position of the bird or birds seen. Generally this will be a single six-letter bird code, but it may be a pair of codes describing a hybrid or species pair.

  The bird ID group for a hybrid has this form.

  ```
  ab6^alt
  ```

  The *ab6* and *alt* codes are the six-letter codes for the probable parents of the hybrids. For example, this group describes a probable hybrid of American Wigeon and European Wigeon:

  ```
  amewig^eurwig
  ```

The bird ID group for a species pair has this form:

```
ab6|alt
```

The *ab6* and *alt* codes are the six-letter codes for the two forms that the observer could not distinguish. For example, this group describes a bird that might have been either a Dusky Flycatcher or a Hammond's Flycatcher:

```
dusfly|hamfly
```

For a single code, the code will appear as the `ab6` attribute of the `taxon-group`. For hybrids and species pair records, the two codes will appear as the `ab6` and `alt` attributes, and the "`^`" or "`|`" will appear as the `rel` attribute of the `taxon-group`.

- If the record is considered notable, an exclamation point "`!`" follows the bird ID group.
- Next are zero or more *suffix groups*, as defined below. If there are multiple suffix groups, each group becomes an `floc` child element of the record's `form` element.
- If there is information to be added later while editing the XML output, a suffix "`*`", called the *long flag*, is added. This forces any output `form` or `floc` elements to be represented as a pair of tags, rather than as an empty tag. This makes it easier to add content to the record later during direct editing of the XML.

Here is the general form of a suffix group, where square brackets indicate an optional element.

```
count-group age-group sex-group status-group
```

count-group
: This group indicates the number of individuals seen. For permissible values, refer to the definition of the `count` attribute of the `age-sex-group` pattern in the schema. Examples: `1`; `57`; `8-10`; `6+`; `40-`; `#`.

age-group
: Any of the permissible `age` attributes of the output: "`a`", "`i`", or "`p`" (for "female or immature").

sex-group
: An optional sex code, "`m`" or "`f`".

status-group
: A code indicating the status of the sighting.

`?`	Questionable identity; outputs an attribute "`q='?'`".
`=`	Not countable under American Birding Association rules; outputs an attribute "`q='-'`".

Here are some examples of complete bird groups and their interpretation.

`rinduc`	One or more Ring-necked Ducks.
`cangoo12`	Twelve Canada Geese.
`whfibi#`	Two or more White-faced Ibis.
`heptan!`	Hepatic Tanager, notable record.
`westan1m`	One male Western Tanager.
`haiwoo1-2`	One or two Hairy Woodpeckers.

`yehbla!4+*`	Four or more Yellow-headed Blackbirds, a notable record. The "*" forces the output `form` element to be rendered as a pair of tags, rather than an empty tag, so that photo links or other information can be added directly to the XML later.
`gamqua3a15+i`	Three adult Gambel's Quail and fifteen or more chicks. The output will contain a `form` element with two `floc` children for the different age classes.
`dusfly|hamfly3+`	Three or more *Empidonax* flycatchers, each of which might have been either a Dusky Flycatcher or a Hammond's Flycatcher.

4.5. A small, complete example file

Here is an example of a complete input file.

```
!nm 2008-04-02 @SocCo Socorro Co.
@NMT Socorro: New Mexico Tech
@home Socorro: 507 Fitch
mouchi*
@SAnt San Antonio
cantow
@home
cubthr

!nm 2008-04-04 @Soc Socorro
@home Socorro: 507 Fitch
casfin1m2f
@Terry Socorro: New Mexico Tech: Terry Lake
amecoo10 rudduc1f barswa rinduc1f1m
@NMT Socorro: New Mexico Tech
belkin1 grtgra1m amecoo2
@Speare Socorro: New Mexico Tech: Speare Hall
houfin* saypho
```

This file describes two field days: April 2 and April 4, 2008. The April 2 section describes four location codes. Code `@SocCo` is the day location, because all records were within Socorro County. Note that the last record, a Curve-billed Thrasher, is preceded by a back-reference to code `@home`, defined on the third line of the file.

Here is the XML produced from this example file:

```
<note-set period="March 2008">
  <!--================================================================-->
  <day-notes date="2008-04-02" day-loc="SocCo" state="nm">
    <day-summary default-loc="SocCo">
      <loc code="SocCo" name="Socorro Co."/>
      <loc code="NMT" name="Socorro: New Mexico Tech"/>
      <loc code="home" name="Socorro: 507 Fitch"/>
      <loc code="SAnt" name="San Antonio"/>
    </day-summary>
    <form ab6="mouchi" loc="home">
    </form>
    <form ab6="cubthr" loc="home"/>
    <form ab6="cantow" loc="SAnt"/>
  </day-notes>
```

```
  <!--====================================================================-->
  <day-notes date="2008-04-04" day-loc="Soc" state="nm">
    <day-summary default-loc="Soc">
      <loc code="Soc" name="Socorro"/>
      <loc code="home" name="Socorro: 507 Fitch"/>
      <loc code="Terry" name="Socorro: New Mexico Tech: Terry Lake"/>
      <loc code="NMT" name="Socorro: New Mexico Tech"/>
      <loc code="Speare" name="Socorro: New Mexico Tech: Speare Hall"/>
    </day-summary>
    <form ab6="rinduc">
      <floc loc="Terry" sex="f" count="1"/>
      <floc loc="Terry" sex="m" count="1"/>
    </form>
    <form ab6="rudduc" loc="Terry" sex="f" count="1"/>
    <form ab6="amecoo">
      <floc loc="Terry" count="10"/>
      <floc loc="NMT" count="2"/>
    </form>
    <form ab6="belkin" loc="NMT" count="1"/>
    <form ab6="saypho" loc="Speare"/>
    <form ab6="barswa" loc="Terry"/>
    <form ab6="grtgra" loc="NMT" sex="m" count="1"/>
    <form ab6="casfin">
      <floc loc="home" sex="m" count="1"/>
      <floc loc="home" sex="f" count="2"/>
    </form>
    <form ab6="houfin" loc="Speare">
    </form>
  </day-notes>
</note-set>
```