# Soil Data Mart Report Manager

# Conventions for Soil Data Mart Reports

## *DocBook XML*

Reports produced with the Soil Data Mart Report Manager use an XML structure, rather than the plain text produced by older NASIS reports. There are new commands in the report language for producing XML, which allow for a range of styles from a very informal data dump to a publication quality product. The language does not enforce specific XML conventions, so the output is determined by your choice of XML elements and attributes[1].

This document describes the conventions for reports to be used in the Soil Data Mart (SDM) and Web Soil Survey. Because these reports are displayed on a public web site and can be included in a generated soil survey document, they must follow specific conventions. In particular, they use the DocBook XML standard to allow interchangeability between several systems. A DocBook reference manual is available at http://www.docbook.org/tdg/en/html/docbook.html.

Within the world of DocBook XML we are using just a subset. Since SDM reports are intended for use within a larger soil survey document, the outermost element of a report is the <section>[2]. That <section> uses the attribute label=”SoilReport” to identify it. Furthermore, the DocBook standard requires that every <section> must include at least a <title> element. All of these requirements can be met by including a standard HEADER block in a report script, such as:

```
HEADER
   ELEMENT OPEN '?xml version="1.0" encoding="utf-8"?'.
   ELEMENT OPEN '!-- DOCTYPE section SYSTEM
 "http://www.docbook.org/xml/4.3/docbookx.dtd" --'.
   ELEMENT OPEN "section" ATTRIB ("label", "SoilReport").
   ELEMENT "title" reporttitle.
   IF reporthdrtext!="" ELEMENT "para" ATTRIB ("label", "headnote")
 reporthdrtext.
END HEADER.
```

[1] An XML **element** is a unit of an XML document consisting of opening and closing tags and usually some content between the tags. A **tag** is a name and optional attributes between angle brackets, such as <para>. An **attribute** is a name followed by an equal sign and a value and is placed inside the opening tag for an element, such as <para class=”mapunit”>. The closing tag for an element is signified by a slash before the element name, like </para>. The complete element would be: <para class=”mapunit”>Alpha silt loam 0 to 3 percent slopes</para>

[2] In the narrative we use the XML convention of angle brackets to designate an element, such as <section>. In examples of report scripts the element names are in quotes, using the report language syntax.

In the above example the line numbers are for reference and are not part of the report script. Notice the syntax of the ELEMENT command. If the word ELEMENT is followed by OPEN or CLOSE, the output will contain only the opening or closing tag of the element. If not, a complete XML element with opening and closing tags is produced. If ELEMENT OPEN is used, there must be a corresponding ELEMENT CLOSE somewhere in the report script (with exceptions as noted below).

Line 2 produces a special tag to identify this file as XML. The question mark indicates that it is not an actual element and does not require a closing tag. This line should be included verbatim.

Line 3 is similar to line 2 in that it identifies the file as DocBook, and it requires no closing tag. However, the tag begins with the sequence !-- and ends with --, which indicates a comment in XML and HTML. Therefore, this line is just a note and has no effect. It can be eliminated but we recommend leaving it in for documentation purposes.

Line 4 is the beginning of the outermost section, as described above. The report language syntax requires that there be quotes around the tag name (“section”) as well as the attribute name (“label”) and value (“SoilReport”). An element can have more than one attribute, by simply adding more ATTRIB ( ) specifications.

Because line 4 only opens the <section>, there must be a closing tag as well. Normally this is handled using a SECTION WHEN AT END block in the report script, such as:

```
8) SECTION WHEN AT END
9)     DATA
10)     ELEMENT CLOSE "section".
11) END SECTION.
```

Line 5 is a complete element definition and produces the required <title> for the <section>. In SDM reports the title of the outermost section is the name of the report or table. This example prints the contents of a variable named `reporttitle`, which will be explained later.

Line 6 is used to print the “head note” for a soil report. Because not all reports have head notes, there is an IF condition in front of this element that says, print only if the `reporthdrtext` is not empty. This element is a <para>, which in DocBook stands for a paragraph, and it uses a “label” attribute of “headnote”. This is a formatting instruction which says to use the paragraph style of “headnote” for the enclosed text. In this way the standard font and layout for soil report head notes is applied automatically.

## *XML Elements Used in Reports*

After the HEADER block the report script will contain a number of SECTION blocks to specify the contents of the report. The ELEMENT command is the main output formatting control and is used either in a TEMPLATE or directly in a SECTION. This is similar to the way the AT command was used in older NASIS reports.

The ELEMENT command can contain a list of data fields, each of which can have a Tag, Valuetag, and Attributes. Tags and Valuetags generate complete XML elements that are placed between the beginning and ending tags of the main element. This creates a structure where the outer element can contain data or other elements. The difference between a Tag and a Valuetag appears when displaying a report variable that has multiple values. A Tag surrounds the entire set of values for the variable, while the Valuetag surrounds each individual value of the variable.

Some examples will illustrate the way ELEMENT can be used.

```
12) ELEMENT “title” “Mapunit Description”.
```

This is the simplest case, consisting of an element name and a string of data. It translates to the following XML:

```
<title>Mapunit Description</title>
```

```
13) ELEMENT “col” ATTRIB (“width”, “3*”).
```

An element can have attributes as well as data. Attributes become part of the opening tag for the element. In this case, the element has no data so the opening and closing tags get condensed into a single tag:

```
<col width=”3*” />
```

```
14) ELEMENT “tr” ATTRIB (“class”, “heading”)
       FIELD TAG “td” ATTRIB (“class”, “begindatagroup”),
       FIELD TAG “td” ATTRIB (“class”, “enddatagroup”).
```

This example shows the use of TAG to create elements within an element. Each of these can have attributes of their own. The first ATTRIB goes with the <tr> element, and the others go with their adjacent <td> elements.

Notice the syntax for this example. The content of an element follows the element name and optional ATTRIB. In this case the content is two FIELDs, so this example would have to be inside a TEMPLATE. The keyword TAG follows FIELD to indicate that the field will be surrounded by tags to create a <td> element. The comma separates the two fields in the <tr> element.

If this example appeared in a TEMPLATE, there would have to be a USING statement to specify the data to be printed. This looks like a heading, so the data might be “Plant Name” and “Pct. Composition”. Then the XML would be:

```
<tr class=”heading”><td class=”begindatagroup”>Plant Name</td>
<td class=”enddatagaroup”>Pct. Composition</td></tr>
```

```
15) ELEMENT “tr” ATTRIB (“class”, shading)
       hzname TAG “td” VALUETAG “para”,
       hzdept_r TAG “td” VALUETAG “para” ATTRIB (“role”, “number”).
```

This example prints data from variables that have multiple values. We can assume that the query in this report used AGGREGATE by component and specified NONE as the aggregation type for the horizon names and depths (hzname and hzdept_r). The TAG “td” places each of these variables in a <td> element, which represents a table cell. Then each individual value is placed in a <para>, which stands for paragraph and means that each will appear on a separate line of output. Thus the horizon names and depths will be listed as side by side columns, although each column is actually within a table cell. This is a common style for soils reports.

Notice the attributes in this example. The <tr> has an attribute value that is not in quotes. It refers to a variable named shading, which would have been created in a DEFINE statement to hold some appropriate class attribute, such as “even” or “odd” (see table of attributes below). Also, the hzname has no attribute, so it gets the default format, which is simply text left justified in the cell.

Assuming some arbitrary data for shading, hzname and hzdept_r, this would create the following XML. This has been indented to make it more readable, but the indentation is not part of the actual report output and has no meaning:

```
<tr class=”odd”>
  <td>
    <para>A</para>
    <para>B</para>
    <para>C</para>
  </td>
  <td>
    <para class=”number”>0</para>
    <para class=”number”>15</para>
    <para class=”number”>75</para>
  </td>
</tr>
```

```
16) ELEMENT “variablelist” VALUETAG “varlistentry”
       hzname TAG “term”,
       hzdept_r TAG “listiem” VALUETAG “para”.
```

This example shows the same data as example 15 in a semi-tabular format rather than in a table. It uses the DocBook element <variablelist> which can be used to create bullets or numbered lists, depending on the final style choices. In the report script we do not need to be concerned with the formatting details, only the structure.

A <variablelist> contains a number of <varlistentry> elements, each of which contains a <term> and a <listitem>. The <listitem> must contain some other element, in this case a <para>. To accomplish this there has to be a VALUETAG at the element level. This means that the <varlistentry> element is produced for each set of values within the element. An additional VALUETAG on the hzdept_r produces a <para> around the horizon depth. The resulting XML is thus four levels deep:

```
<variablelist>
  <varlistentry>
    <term>A</term>
    <listitem>
      <para>0</para>
    </listitem>
  </varlistentry>
```

```
      <varlistentry>
        <term>B</term>
        <listitem>
          <para>15</para>
        </listitem>
      </varlistentry>

      <varlistentry>
        <term>C</term>
        <listitem>
          <para>75</para>
        </listitem>
      </varlistentry>
    </variablelist>
```

## Elements used in tables

The DocBook standard includes the HTML standard for creating tables, which will be familiar to most people who have created web pages. The basic elements of a table are listed here. All of them are needed for a complete SDM report, but for a report that is not for publication many of the elements can be left out.

- <table> encloses the whole table.

- <title> is for a title that is printed above the table. In SDM reports this is the name of the soil survey area. If the report uses national mapunit symbols there is no survey area and the table title is omitted.

- <col> specifies attributes for a column in the table. If used, there must be one <col> element for each column. In SDM reports this is used to specify a relative width for the column.

- <thead> is for the table header. Rows within a <thead> receive special formatting so they look like a heading. They are also repeated at the top of each report page when the report is converted to PDF.

- <tbody> is the body of the table and occurs after the <thead>.

- <tr> is a row of a table. It can be used within a <thead> or a <tbody> and it contains one or more <td> elements for the data in the row.

- <td> is table data, sometimes called a cell of a table. It must be within a <tr> element.

- <para> is a paragraph that can be included in a <td>. It is not required, since data can be included within a <td> directly. The <para> is used in SDM reports to identify the type of data so that standard formatting styles can be applied.

  Note that <para> is a DocBook element for paragraph, and the corresponding HTML element is <p>. SDM reports use the DocBook convention, but if you are creating a report with pure HTML output, the <p> should be used.

## Elements used in non-table reports

Reports that are not formatted as a table include the various Map Unit Description reports, the Interpretation Description, and similar reports. Although there is no <table> element there is an internal structure to these reports, and the <section> element is generally used to represent it. The DocBook standard allows sections to be included within sections. So, for example, if a report is organized by survey area there will be a <section> for each new area name. A Map Unit Description report will have a <section> for each mapunit within the area, and a variety of <section>s within the mapunit. Each <section> must have a <title>, which is used as a heading for the section when it is displayed.

Data in these reports is typically displayed in a list (semi-tabular) format. The DocBook <variablelist> element is used to create a list structure, which can then be displayed in various ways. Example 16 above provides an explanation of the use of <variablelist>.

Reports that contain large blocks of text will use the <para> element to enclose the text. The way you expect the text to be stored in the database will influence the attributes applied to the <para>. If the text is simply a single paragraph no special attributes are needed. If the text contains line breaks, tabs or spaces that need to appear in the output, ATTRIB ("role", "preservenewlines") is needed. If the text contains XML markup that needs to be preserved in the output, use the SPECIAL option, as in:

```
textdata TAG "para" SPECIAL.
```

## Attributes used in all elements

Attributes describe the type of data or the role played by an element in the overall report structure. This information is used when formatting the report for display, and it can be used in different ways depending on the display format. For example, the same report might appear in a web site and in a PDF document meant for printing. This separation of the functional description of a report and its display formatting allows one report script to serve multiple purposes. However, only certain attributes are recognized by the formatter, and unrecognized attributes are ignored. So the conventions listed here must be followed closely to produce reports with a consistent appearance.

| Element | Attribute Name | Attribute Value | Meaning |
|---|---|---|---|
| "section" | "label" | "SoilReport" | Required to identify the outermost section of a soil report |
| | | "Survey_Area" | Data for one survey area. The title of the section is the survey area name. Used in reports that don't have tables. |
| | | "Map_Unit_Description" | Data for one mapunit in a map unit description report. |
| | | others | The Map Unit Description report has labels to signify the type of data in each section, but they are currently ignored by the |

| | | | |
|---|---|---|---|
| | | | formatting program, |
| ”title” | ”role” | ”suppressTitle” | Do not display a title for the current section. Although the <title> element is required in a <section>, this will suppress display of the title. |
| ”table” | ”orient” | ”land” | Table is wide and should be displayed in landscape orientation. |
| “col” | “width” | “n*” | Relative column width expressed as a number followed by an asterisk. A column with width “3*” is 3 times as wide as a column with width “1*”. The exact width of a column depends on the overall width of the table and width of the other columns. |
| ”tr” | ”class” | ”mapunit” | A table row that begins the data for a map unit. |
| | | ”even” | A table row that can be shaded in alternating colors to improve readability. Alternates with “odd”. |
| | | ”odd” | A table row that can be shaded in alternating colors to improve readability. Alternates with “even”. |
| | | ”interpdata” | A table row containing data about an interpretation in the Survey Area Data Summary report. |
| | | ”units” | A table row containing units of measure. It is a type of subheading for the table |
| ”td” | ”rowspan” | a number | Number of rows in the heading occupied by this column. Part of standard HTML tables. |
| | “colspan” | a number | Number of columns in the heading occupied by this cell. Part of standard HTML tables. |
| | “class” | “begindatagroup” | Cell is the first of a group of cells that are set off visually by a heavier vertical border on the left edge. |
| | | “enddatagroup” | Cell is the last of a group of cells that are set off visually by a heavier vertical border on the right edge. |
| | | “datetime” | Cell contains a date/time field that should be formatted according to the date |

|  |  |  |  |
|---|---|---|---|
|  |  |  | conventions for the report. |
| "para" | "role" | "mu-name" | Content is a mapunit symbol or name. |
|  |  | "comp-name" | Content is a component name. |
|  |  | "number" | Numeric data, normally displayed right justified in a cell. |
|  |  | "class-name" | Non-numeric data, such as a class name, normally displayed centered in a cell. |
|  |  | "hang-list" | Multiple values of character type data which are displayed in a vertical list with hanging indents |
|  |  | "preservenewlines" | Content is a text field that may include newlines, tabs, and significant spaces. Normally this "white space" is removed. This attribute will preserve the layout of the original text. |

## *Parameters for Reports*

A general report script can use any parameters, but reports for the SDM and WSS must use the parameters that those systems supply. In general, a user in SDM or WSS can choose either a full soil survey area or a list of map units to report on. These choices will be passed to the report script. Other parameters are used in specific reports. The parameter names and types must be defined as in the list below. The report script can also define a prompt for each parameter, which will appear in the user interface for entering parameter values when applicable.

**#PARAMETER areasym CHAR.**
The area symbol for the requested survey area. If selecting by map unit, this will be null.

**#PARAMETER mukeys MULTIPLE CHAR.**
The map unit keys for the selected map units. If selecting by survey area this list will be empty.

**#PARAMETER includeminor BOOLEAN.**
Set to 1 if minor soils are to be included in the report, or 0 if they are to be excluded.

**#PARAMETER useNationalMapunits BOOLEAN.**
Set to 1 if national mapunit symbols (mukey) are to be printed, or 0 if survey based mapunit symbols (musym) are to be used.

**#PARAMETER crops MULTIPLE CHAR.**
The list of crop names selected for inclusion in a crop yield report.

The last example uses what is called a "dynamic choice list", meaning that the list of choices depends on the area of interest selected by the user. There is currently a limited set of parameters that can be used as dynamic choice lists. In place of “crops” the following parameter names can be used:

**cocrops:** Crop names from the Component Crop Yield table that have non-null yield data.

**mucrops:** Crop names from the Mapunit Crop Yield table that have non-null yield data.

**interps:** Names of interpretations (rules) exported with the selected survey area(s).

**sainterps:** Same as **interps** but allows more than 3 selections.

**cotextkinds:** Component text kind and category.

**mutextkinds:** Mapunit text kind and category.

**nontechdesccats:** Mapunit text categories where text kind is “nontechnical description”.

**counties:** County names found in the Legend Area Overlap table for the area of interest.

## Using parameters in a report query

Most of the parameters listed above are used in the report queries to control the selection of data. In one case, “useNationalMapunits”, the parameter is also used in the body of the report to alter parts of the output. In the queries, some unusual SQL features are used. This is an example from a crop yield report that uses most of the parameter options:

```
EXEC SQL select areaname, musym, museq, mapunit.mukey muiid, muname,
    compname, comppct_r, localphase, component.cokey,
    cropname, yldunits, irryield_r, nonirryield_r
    from legend, mapunit, component, outer cocropyld
    WHERE join legend to mapunit
    and join mapunit to component
    and join component to cocropyld
    and ( areasymbol=$areasym OR mapunit.mukey in ($mukeys) )
    and ( majcompflag = "Yes" OR 1 = $includeminor )
    and cropname in ($crops)
    ORDER BY case when $useNationalMapunits =0 then areaname else "0" end,
      case when $useNationalMapunits =0 then museq else
      cast(mapunit.mukey as int) end,
      cropname;
    AGGREGATE ROWS muiid
    COLUMN yldunits UNIQUE GLOBAL
    CROSSTAB cropname VALUES (crops)
    CELLS irryield_r, nonirryield_r, yldunits.
```

Line 8 uses the “areasym” and “mukeys” parameters to select the map units for the reports. By using OR, it will select either by survey area or by mapunit, depending on which parameter is supplied. Note that when testing this report, if you enter values for both parameters it will select mapunits that meet either criterion, which may be more than you expected.

Line 9 uses the “includeminor” parameter to select minor components. Here OR is used to say, select a component if it is a major component, OR if the includeminor parameter is set to 1.

Line 10 uses the “crops” parameter to select crop names. The parameter is used again in line 16 to specify the crosstab values. Doing this forces the report to include a column for each crop, even if there is no yield data for a crop. It also forces the columns to appear in the order that the crop names are entered in the parameter.

Lines 11 and 12 make use of the “useNationalMapunits” parameter to control the sorting of the report. This is tricky because when survey area mapunit symbols are used the report is sorted by survey area, but when national mapunits are used it is not. The SQL CASE expression is used here, which is similar to the IF expression in a DEFINE statement. Line 11 says to sort by areaname or by”0” depending on whether survey area or national symbols are used. Sorting by a constant value like “0” is just a way of saying, don’t sort on this column.

Line 12 says to sort on museq or mukey. The column museq is the sequence number for mapunits within a legend, so it is used to sort when using survey area map units. To sort on mukey there is a CAST, which forces the mukey to be sorted numerically. The values of mukey are actually numbers but they are stored in the database as characters, so this cast is needed.

## *Script Variables*

A script variable is a special notation used to obtain information about the report script itself from the database. An example is the title for the report, which is stored in the “report” table in the column “reporttitle”. To include this title in the output of a report, use a reference to the script variable REPORT_TITLE, as in:

```
    DEFINE reporttitle INITIAL SCRIPT(REPORT_TITLE).
```

This places the script variable into a normal report variable, reporttitle, which can then be printed as part of the report heading. Using INITIAL in this statement means that the variable reporttitle is set at the beginning of report processing and never changes.

The script variables available for SDM reports are:

**REPORT_TITLE** The report title from the Home tab in the report editor.

**REPORT_HEADER** The text to be used as a head note in the report, from the Home tab in the report editor.

**INTERP_NAME** A list of rule names to be included in an interpretation report, from the Interpretations tab in the report editor.

**INTERP_TITLE** The list of column headings to be used with the corresponding rule in the INTERP_NAME list. This is also entered in the Interpretations tab.

**SCRIPT_ID** The internal ID number for the record in the report table. This might be used to query for data linked to the report in the database.

Using script variables allows a report script to be generalized, so that certain features do not have to be coded into the script. This capability is used for interpretation reports, which use a completely generic script. All standard interpretation reports in SDM use exactly the same script. That script uses script variables to pick up the report name, head note, and list of interpretations from data stored in the report table. To create a state report using custom interpretations you can simply copy an existing interpretation report and change the list of interpretations and titles in the Interpretations tab. You also specify which states will use the report in the Usage tab.